Microcontroller-Based PC Thermometer with Regulated Fan

By

James Christian B. Aspillaga

Reinald Carlo A. Roque

Ishmael Angelo F. Sanchez III

A Design Report Submitted to the School of Electrical Engineering, Electronics and Communications Engineering, and Computer Engineering in Partial Fulfillment of the Requirements for the Degree

Bachelor of Science in Computer Engineering

Mapua Institute of Technology

January 2009

Approval Sheet

Mapua Institute of Technology
School of EE-ECE-CoE

This is to certify that we have supervised the preparation of and read the design report prepared by **James Christian B. Aspillaga, Reinald Carlo A. Roque, Ishmael Angelo F. Sanchez III** entitled **Microcontroller-Based PC Thermometer with Regulated Fan** and that the said report has been submitted for final examination by the Oral Examination Committee.

Prof. Gil Astrophel Orcena Reader	**Engr. Erwin Donavan Ong** Adviser

As members of the Oral Examination Committee, we certify that we have examined this design report, presented before the committee on **December 17, 2008**, and hereby recommend that it be accepted as fulfillment of the design requirement for the degree in **Bachelor of Science in Computer Engineering.**

Engr. Cheryl Mari Isip Panel Member 1	**Engr. Eliza Agpoon** Panel Member 2

Engr. Ronina Caoili
Panel Member 3

This design report is hereby approved and accepted by the School of Electrical Engineering, Electronics and Communications Engineering, and Computer Engineering as fulfillment of the design requirement for the degree in **Bachelor of Science in Computer Engineering.**

Dr. Felicito S. Caluyo
Dean, School of EE-ECE-CoE

ACKNOWLEDGEMENT

We thank the Lord for all the blessings you have given us, for the lives you have given to us. Without your power and your guidance, we would not be able to finish this work.

We would also like to thank our parents, for without them, we would not be where we are today. Our parents were the ones who guided us and provided us our wants and needs. They are our inspirations in finishing this work. Now, we want to return a part of their hard work by means of this project.

We would also like to give thanks for our friends and family members for giving us support in every endeavor we undertake.

TABLE OF CONTENTS

TITLE PAGE	i
APPROVAL SHEET	ii
ACKNOWLEDGEMENT	iii
TABLE OF CONTENTS	iv
LIST OF TABLES	v
LIST OF FIGURES	vi
ABSTRACT	vii
Chapter 1: DESIGN BACKGROUND AND INTRODUCTION	1
The Design Setting	1
Statement of the Problem	2
The Objective of the Design	2
The Significance of the Design	3
The Conceptual Framework	4
The Scope and Delimitation	5
Definition of Terms	6
Chapter 2: REVIEW OF RELATED LITERATURE	10
Chapter 3: DESIGN METHODOLGY AND PROCEDURES	15
Design Methodology	15
Design Procedure	15
Hardware Design	18
Block Diagram	18
Schematic Diagram	19
List of Materials	20
Software Design	22
System Flowchart	24
Prototype Development	25
Chapter 4: TESTING, PRESENTATION, AND INTERPRETATION OF DATA	29
Testing using Multimeter	29
Testing Fan Speed with Digital Tachometer	31

Testing on Inflow Direction of Air versus Outflow Direction of Air 33
Testing on a VOIP server 37

Chapter 5: CONCLUSION AND RECOMMENDATION 41

Conclusion 41
Recommendation 41

References 43

Appendices 44

Appendix A: Source Code 44
Appendix B: PIC16F87x Data Sheet 57
Appendix C: LM35 Data Sheet 65
Appendix D: Installation Manual 78
Appendix E: User's Manual 82

LIST OF TABLES

Table 3.1 List of Materials 20
Table 4.1: Test of DC Fan without Device 30
Table 4.2: Test of DC Fan with Device 31
Table 4.3 Fan Speed and Corresponding Revolution per Minute 33
Table 4.4 Inflow Direction of Airflow 34
Table 4.5 Outflow Direction of Airflow 35
Table 4.6 Outflow Direction of Airflow since Start-Up 36
Table 4.7 Inflow Direction of Airflow since Start-Up 36
Table 4.8 Test on the first few minutes after boot-up at initial CPU temperature of 29ºC 39
Table 4.9 Test on the Server after Reaching Steady State 40

LIST OF FIGURES

Figure 1.1 Conceptual Framework	4
Figure 2.1 Fan Speed vs. Fan Voltage	10
Figure 2.2 Fan Current vs. Fan Voltage	11
Figure 2.3 Linear Fan-Control Circuit	11
Figure 2.4 Thermistor Based Fan-Control Circuit	12
Figure 3.1 Design Procedure Flowchart	16
Figure 3.2 Block Diagram	18
Figure 3.3 Schematic Diagram	19
Figure 3.4 LM35 Sensor	21
Figure 3.5 PIC16F877 Microcontroller	21
Figure 3.6 Cooling Fan	22
Figure 3.7 System Flowchart	24
Figure 3.8 Compiling the Code using MPLAB IDE	27
Figure 3.9 Uploading the Code to the Microcontroller Using IC-Prog	27
Figure 4.1 Testing Prototype with Multimeter	30
Figure 4.2 Testing Fan Speed using Tachometer	32
Figure 4.3 Softphones used for testing in Voice over IP server	38

ABSTRACT

This device is a temperature monitoring device with regulated fan for computers. The purpose of this study is to develop a circuit that adjusts the motor speed of computer cooling fan/s depending on the measured temperature. A thermal sensor is used to measure the temperature of the computer. The device is also integrated with a microcontroller which enables the cooling fan to attain a variable motor speed depending on the measured temperature. Varying the motor speed of the fan results to lesser power consumption. The researchers used the multimeter to test the conservation of power consumption and a digital tachometer to test the varying motor speed of the fan. Findings of this study showed that using a thermal sensor, microcontroller and other components, a computer cooling fan can vary its motor speed and reduce power consumption.

Keywords: Thermal sensor, Microcontroller, Computer

Chapter 1

DESIGN BACKGROUND AND INTRODUCTION

1. The Design Setting

Over the past ten years, computers have increased their heat output considerably. Even a cursory glance at a modern server reveals the huge amount of effort that the designers had to take to keep the computer cool. Not only do the processors have massive heat sinks, but the server also gets fully loaded with fans. In extreme cases, liquid cooling is necessary. The bad news is that things are getting worse. Processor speeds are going up; heat generation is climbing too.

Computers are now very important to our daily lives. They are used in businesses, for educational purposes, and almost everything can be done using a computer. That is why proper care for these computers is very essential. Some people use their computers for a long period of time. One problem that may occur is overheating of the computer resulting to loss of valuable data and even worse, destruction or cutting short the life of the computer system's components. For servers, temperature monitoring is an issue; if the server becomes too hot, it can cause to down time in servers. The cost of a catastrophic server failure can be considerable. Much costs would be met if the server goes down. There is the cost of replacement, loss

of e-commerce business, loss of customers details, waste of staff time and all the other associated costs.

2. Statement of the Problem

Computer equipment ages faster when it gets hot. In general, computer operates more reliably and has a longer life in cooler conditions. The effects of prolonged running at high temperatures can be unpredictable and are not always characterized by catastrophic failures.

For individual machines, in domestic or small office conditions the internal fans and cooling mechanism are usually sufficient to keep the temperature within safe. But if the machine is used in a business wherein the uptime of the machine is critical and is used for a long time. Temperature monitor is important.

Temperature monitoring is essential for CPU especially if it is used for server. Failure to monitor temperature could result to unnoticed heat build-ups in CPU that can reduce the life of components, decrease reliability, cause untold problems and expense.

3. The Objective of the Design

The objective of this project design is to measure the present temperature inside the computer and will be displayed on its casing for easier temperature monitoring and to create a prototype of a computer

thermometer that can be placed on the central processing unit as an additional hardware. Using the PIC microcontroller, the team would design an external display of the temperature on the CPU. Through this external display, temperature monitoring will be easier especially on servers that are sharing to one monitor since it will only need one look in the CPU to know the temperature. Another objective of our design is to determine whether which airflow is better, either inflow direction of air or outflow direction of air.

4. The Significance of the Design

The design of the device would be very helpful to computer users because it can help prevent overheating of their computer through proper monitoring of its temperature and applying a temperature dependent fan to maintain desirable temperature. The device will be placed inside the computer. The device will measure the temperature inside the CPU and introducing a fan into the system to exhaust the hot air around the chassis and the heat-generating components. When it becomes too hot, the fan's speed would be faster and if the temperature is on the average, the fan's speed would be normal just to maintain the desirable temperature. Automatic fan speed control without the need for host intervention is particularly useful in PC applications for a number of reasons. Once configured, it enables the system to react to temperature changes and ensures that the fan will run

only as fast as required for any given temperature, minimizing the power consumption and reducing the noise level generated by the fan.

5. The Conceptual Framework

The device is microcontroller-based to monitor the temperature and regulate the speed of the fan. It contains all the programs needed to have the accurate reading of the thermometer and to control the fan speed. There is also an LCD display that indicates the current temperature inside the casing. Refer to Figure 1.1 for graphical presentation of the conceptual framework.

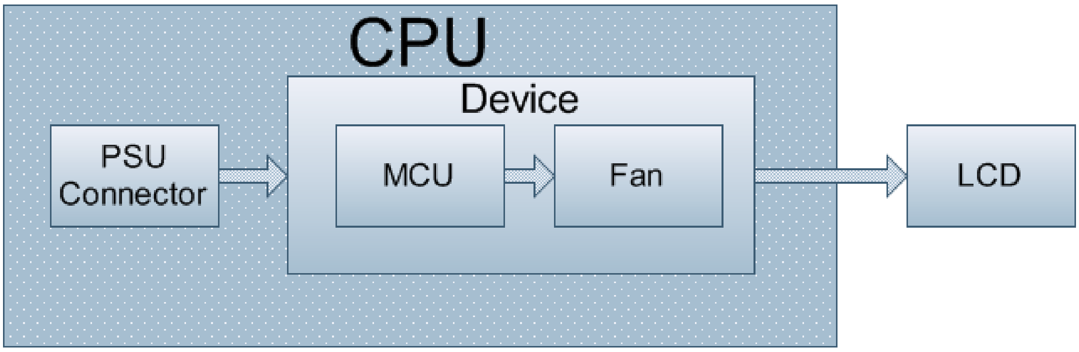


Figure 1.1 Conceptual Framework

6. The Scope and Delimitation

The study was concerned with the development of a design called Microcontroller-based PC Thermometer with Regulated Fan that serves as a cooling and temperature monitoring of a computer. The research study set the scope and delimitation as follows:

1. The prototype can be place in any computer that has CD/DVD ROM rack of the CPU as an additional hardware;
2. The LCD will only display the current temperature inside the casing and corresponding fan speed;
3. The device receives its power from the computer's PSU (Power Supply Unit) that produces 5V and 12V;
4. The sensor can be mounted in any part of the CPU that needs temperature monitoring;
5. The thermally activated fan is included to help maintaining desirable temperature inside the CPU; and
6. The prototype is designed for server use.

The delimitations of the design of the "Microcontroller-based PC Thermometer with Regulated Fan" are as follows:

1. The LCD can only display the current temperature inside the casing in degrees Celsius format;
2. There should be an available connector from the PSU (Power Supply Unit) as a source of power;

3. The casing has to be modified to acquire the full potential of the device;
4. Different model of the internal parts of the motherboard has different tolerable heat; and
5. Different design of CPU has different airflow which can lead to different result if tested with another computer.

7. Definition of Terms

The group encountered some terms that were used throughout the study. These are as follows:

American Standard Code for Information Interchange (ASCII) – a standard for assigning numerical values to the set of letters in the Roman alphabet and typographic characters. *(Microcontrollers: Architecture, Implementation and Programming Boston, McGraw-Hill)*

Capacitor - a passive element designed to store energy in its electric field, the most common electrical components. It is consisted of two conducting plates separated by an insulator (or dielectric). It is an open circuit to dc used extensively in electronics, communications, computer, and power systems. *(Fundamentals of Electric Circuits, 2004)*

Celsius - using or measured on an international metric temperature scale on which water freezes at 0° and boils at 100° under normal atmospheric conditions. *(University Physics)*

Central Processing Unit (CPU) - part of the computer that holds the main components like mother board and video card. This is usually enclosed in a casing. *(Computer Concepts and Fundamentals)*

Current - current, flow of electric charge. The electric charge in a current is carried by minute particles called electrons that orbit the nuclei of atoms. *(Fundamentals of Electric Circuits, 2004)*

Direct Current (DC) - unidirectional flow of electric charge. Direct current is produced by such sources as batteries, thermocouples, solar cells, and commutator-type electric machines of the dynamo type. *(Fundamentals of Electric Circuits, 2004)*

Frequency - the number of times a specified periodic phenomenon occurs within a specified interval *(University Physics)*

Light Emitting Diode (LED) - a semiconductor diode that emits light when conducting current, and is used in electronic equipment, especially for displaying readings on digital watches, calculators, etc. *(Computer Concepts and Fundamentals)*

Linear Regulation - adjusts the dc voltage across the fan using a linear regulator. *(Electronics Design, Strategy, News)*

Microcontroller - single purpose processing units designed to execute small control programs, sometimes in real time. The program is frequently stored on the microcontroller in an area of nonvolatile memory. *(Handbook of Microcontrollers)*

Multimeter - an electronic measuring instrument that combines multiple functions; a combined voltmeter, ammeter and ohmmeter. *(Fundamentals of Electric Circuits)*

Nybble - the computing term for a four-bit aggregation, or half an octet (an octet being an 8-bit byte). As a nibble contains 4 bits, there are sixteen possible values, so a nibble corresponds to a single hexadecimal digit. *(Computer Concepts and Fundamentals)*

PIC - a family of Harvard architecture microcontrollers made by Microchip Technology, derived from the PIC1640 originally developed by General Instrument's Microelectronics Division. The name PIC initially referred to "Programmable Interface Controller", but shortly thereafter was renamed "Programmable Intelligent Computer". *(Handbook of Microcontrollers)*

Programmable Logic Device (PLD) - an electronic component used to build reconfigurable digital circuits. Unlike a logic gate, which has a fixed function, a PLD has an undefined function at the time of manufacture. *(Computer Concepts and Fundamentals)*

Rectifier Diode - a semiconductor device that converts ac into pulsating dc; one part of a power supply. *(Electronic Devices, 2002)*

Resistor - the simplest passive element. It is a device that has the ability to resist the flow of electric current that is measured in ohms. It is usually made from metallic alloys and carbon compounds. (*Fundamentals of Electric Circuits, 2004*)

Thermistor - a resistor whose resistance varies as a function of temperature. Thermistors are used in electrical devices such as thermometers and thermostats that measure, monitor, or regulate temperature. *(Fundamentals of Electric Circuits)*

Transistor - a semiconductive device used for amplification and switching applications. (*Electronic Devices, 2002*)

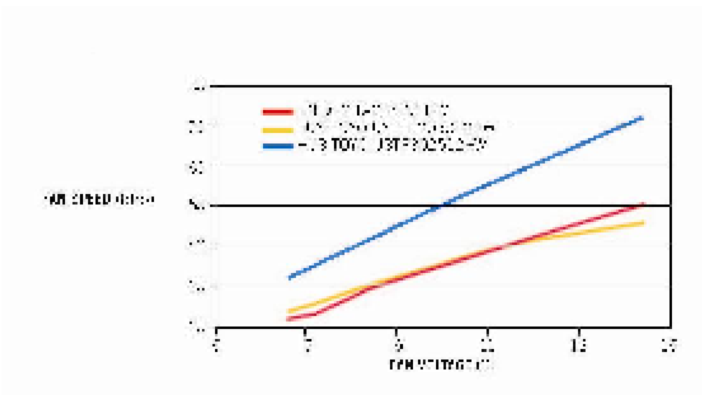
Tachometer - an instrument that measures the speed of rotation of the engine, in revolutions per minute (rpm). *(University Physics)*

Voltage - electric potential expressed in volts *(University Physics)*

Chapter 2

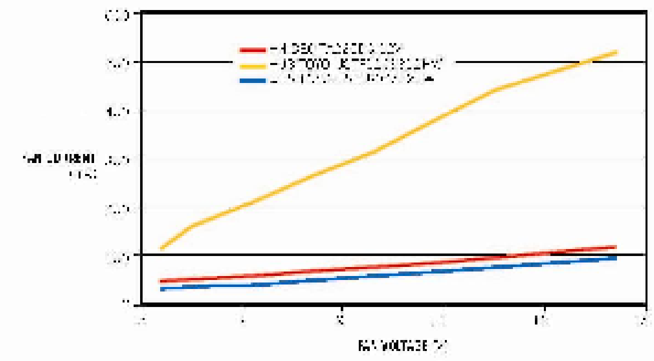
REVIEW OF RELATED LITERATURE AND RELATED STUDIES

In an article entitled “Cooling Down with Fan-Speed Control” on September 28, 2000, Bruce Denmark conducted a study that gave an important contribution to our study which is to develop a device that can display the temperature of computer and maintain a desirable temperature inside the computer. He said that introducing fan-speed control into electronic designs can help in reducing heat problems. He also discussed the different methods in which fan-speed control can be achieved, like Pulse-Width Modulation, Linear Regulation and DC/DC Regulation. In Mr. Denmark’s article, he mentioned that “as the dc voltage applied to the fan varies, its speed and current draw also vary” as portrayed in Figure 2.1. He also mentioned that “the speed and current of the dc fan are directly proportional to the dc voltage applied” as shown in Figure 2.2.



Fan speed, like current, is also proportional to applied fan voltage.

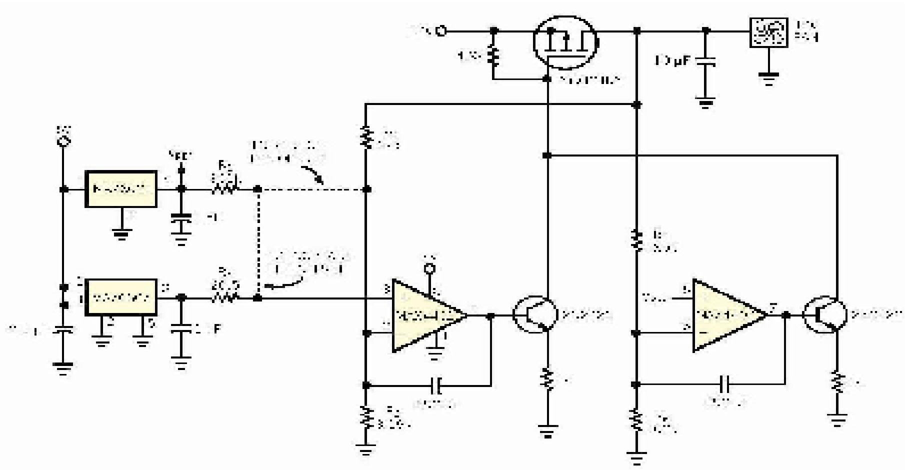
Figure 2.1 Fan Speed vs. Fan Voltage



The fan draws a current that is proportional to applied fan voltage but varies from fan model to model.

Figure 2.2 Fan Current vs. Fan Voltage

Furthermore, an article by Jim Christensen, which is entitled "Circuit Generates Fan-Speed Control" on March 21, 2002, states that "linear regulation" can be used to control the fan-speed, which he showed in his circuit in Figure 2.3, as for our design "pulse-width modulation" is used. This study gave the researchers an alternative in producing a fan-speed control circuit. The group also considered this study to gather relevant information like the components that he used for the circuit.



This circuit delivers a continuous and linear fan-control voltage that is proportional to temperature.

Figure 2.3 Linear Fan-Control Circuit

Moreover, John Guy in his article entitled "Circuit Provides Efficient Fan-Speed Control" on March 4, 2004 made a study similar to our design which is a circuit that provides efficient fan-speed control. John Guy created a circuit that has the same objective of our design. He implemented his logic to control fan-speed by using a thermistor as a temperature monitor. The thermistor produces voltage that is proportional to the temperature. The circuit, as shown in Figure 2.4, uses a thermistor, gave the researchers' a possible alternative as a temperature sensor. The circuit included multiple power outlets that allowed multiple fans to be connected. This study gave the group possible improvements that can be implemented on the design.

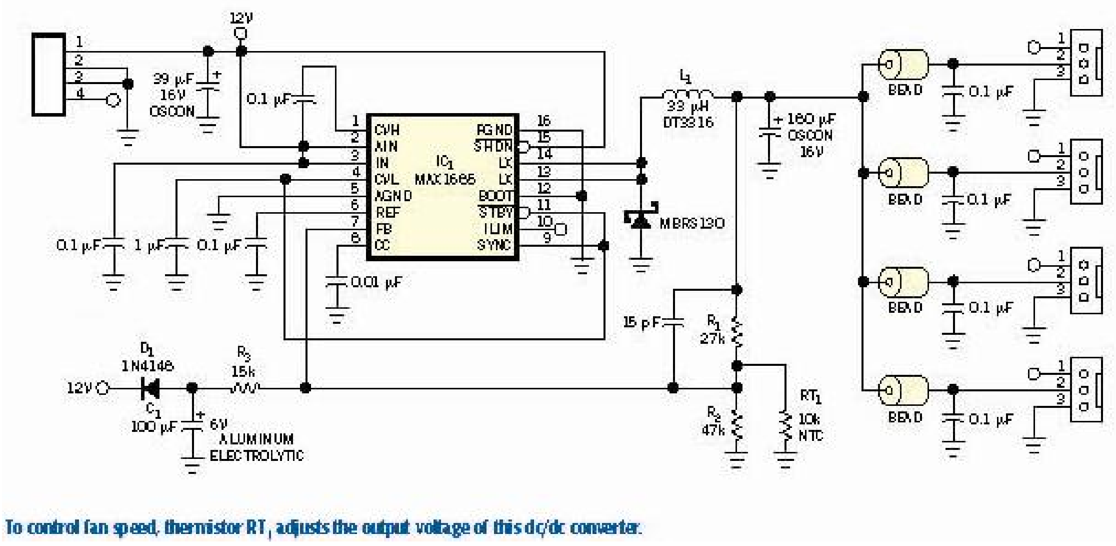


Figure 2.4 Thermistor Based Fan-Control Circuit

Additionally, an article written by Ken Gay (2007) entitled "Understanding Closed-Loop Fan Speed Control", showed two types of fan speed control variation which are the Closed Loop Variation and Open Loop Variation. His article mostly discussed closed loop variation, what are its benefits and

advantages. This gave the group the knowledge that there were two types of variation. The group also learned that closed loop variation uses a tachometer in order to adjust its speed.

In Mr. Darrin Vallis' article entitled "Closed-Loop Fan Control at System Level", (2004) he discussed briefly the disadvantages of an open loop and how closed loop can be better in comparison. This article gave the group a wider knowledge on the advantages of a closed loop. The article also discussed that a tachometer feedback is used in a closed loop variation. Mr Vallis also discussed in his article what tachometer feedback is and how it works.

In addition, Myke Predro (1997), in his book entitled "Handbook of Microcontrollers" discussed *Motor Control* and *LCD Control*. The article discussed the different methods that can be used to control motors. The article discussed that the easiest way to control motors is to switch them on and off. This study was used by the group since the project involved is controlling the fan speed depending on the temperature. The article suggested that controlling the motor speed is normally done by pulsing the control signals in the form of a pulse wave modulated signal, which is usually called Pulse Width Modulation. The group used the knowledge gained from the article about pulse width modulation and other methods on controlling motors in the design of the project. Also, the article on LCD control focused on the discussion on how to use an LCD. The article also discussed the basic pin configuration usually used. The article was used by the group to gather information on LCD, how they work and how the LCD can be

connected to the microcontroller. The article is very helpful to the researchers since the article was very in-depth in its discussion.

Lastly, Myke Predro (2000) in his article entitled "Programming and Customizing PIC Micro Microcontrollers Second Edition", discussed pulse-width modulation (PWM) in detail. It explained that using pulse-width modulation is the best way to handle analog voltages and that PICmicro MCU does not handle voltages very well. The study also discussed that analog signals should not be used for data transfer. His article was useful to our study since pulse-width modulation is also the method we used in our design.

Chapter 3

Design Methodology and Procedures

Design Methodology

Constructive research was used as the design methodology. The design procedure part will provide further explanation of the method and a step by step procedure will be discussed. The research started by identifying the problem. Followed by collecting information relevant to the study, formulating a plan or draft and gathering the materials needed. Figure 3.1 (Design Procedure Flowchart) shows the procedure in developing the design. On the succeeding pages, Figure 3.2 shows the block diagram of the PC thermometer with regulated fan.

Design Procedure

The design follows the constructive research method as stated from above. On figure 3.1 (Design Procedure Flowchart) it will clarify the details step by step. The design has four parts: first is identifying the problem, second is collecting information relevant to the study, the third is formulating a plan or draft and the last part is gathering the materials needed.

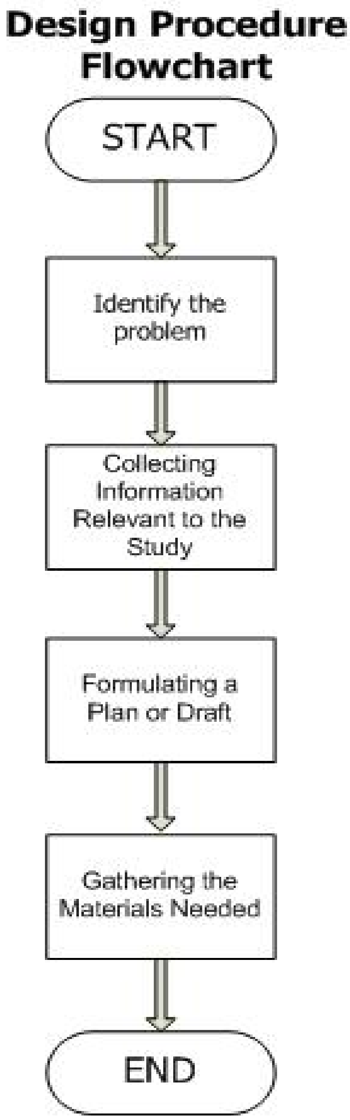


Figure 3.1 Design Procedure Flowchart

1. Identify the problem – the first thing to do in a project is to identify the problem to be solved. In our design, the problem we identified is that failure to monitor temperature could result to unnoticed heat build-ups in CPU that can reduce the life of components, decrease reliability, cause untold problems and expense.

2. Collecting Information Relevant to the Study – this part of the study is where research is done to gather important details that can help accomplish the project to be made.
3. Formulating a Plan or Draft – this part of the study is where the researchers prepare the objectives and tasks to be done needed to finish the project.
4. Gathering the Materials Needed – this is the last part of the design procedure of our project in which all the necessary tools and equipments needed are gathered. This is where all the hardware components and the software to be used will be determined.

a. Hardware Design

1. Block Diagram

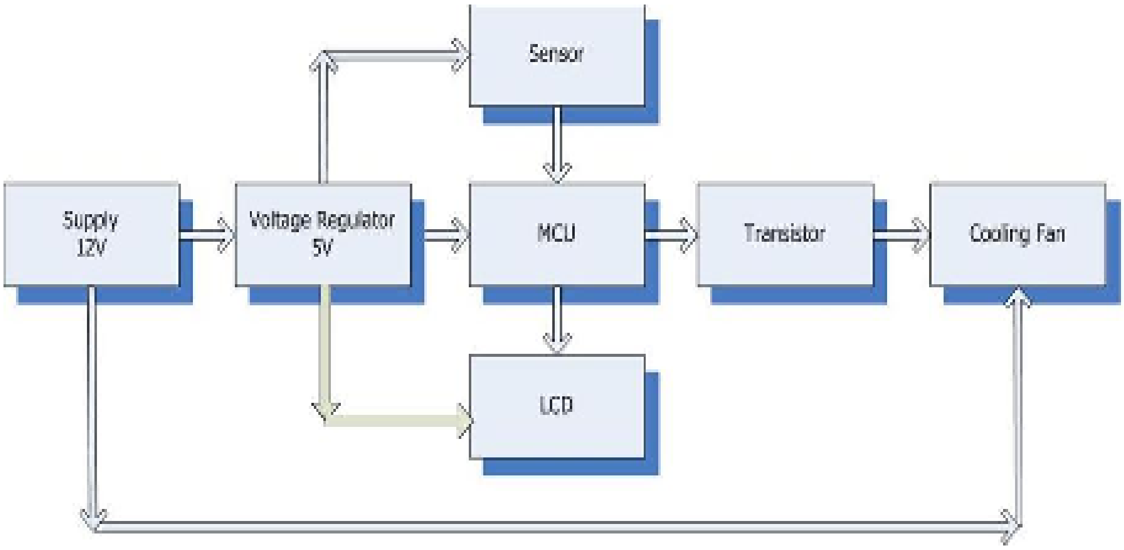


Figure 3.2 Block Diagram

In Figure 3.2, as shown above, the power supply of the CPU will be used as the main source of the hardware. The 12 V will supply the cooling fan and the 5 volts is for the sensor and the microcontroller. When the sensor read temperature it will send an analog signal to the microcontroller. The microcontroller will convert the analog signal into its digital value in order for it to be displayed in the LCD and at the same time it will send a signal to the transistor in order to trigger the cooling fan.

2. Schematic Diagram

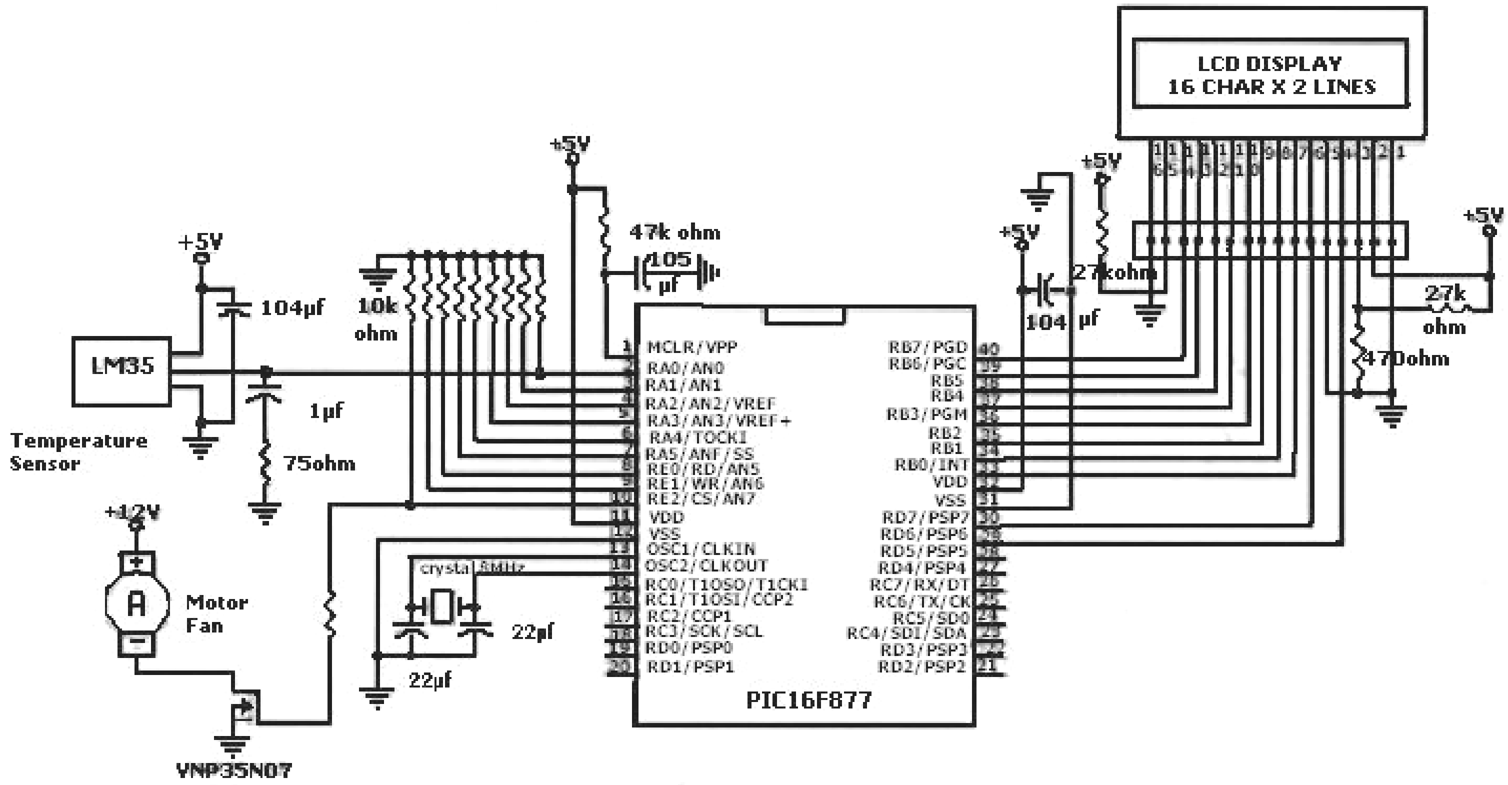


Figure 3.3 Schematic Diagram

3. List of Materials

Material	Quantity
PIC16F877	1
LM35	1
LCD 2X16	1
8 MHz Crystal Oscillator	1
VNP35	1
104 Multilayer Capacitor	1
105 Multilayer Capacitor	1
¼ watt resistor	4
1000 microfarad/16v electrolytic capacitor	1
22picofarad ceramic capacitor	2
Blower Fan	1
Heat Sink	1
2 pins Terminal Block	1
8 pins Connector	1
10k array resistor	1

Table 3.1 List of Materials

The essential components of the design are the thermal sensor, microcontroller and the cooling fan. The thermal sensor which is the LM35 Sensor (Figure 3.4) directly measures the temperature inside the computer. The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. Other materials used in the project are shown in Table 3.1. With the different temperatures measured, the microcontroller which is PIC16F877 (Figure 3.5) performs various functions the sets or regulates the speed of the cooling fan. The obtained temperatures and the fan speed are displayed using the LCD panel. A microcontroller or Programmable Interface Controller (PIC) is a functional computer system-on-a-chip. In addition to the usual arithmetic and logic elements of a general purpose microprocessor, the microcontroller integrates

additional elements such as read-write memory for data storage, read-only memory for program storage and input / output peripherals. The cooling fan (Figure 3.6) is used to eliminate the heat produced by the computer. The speed of the fan depends on the measured temperature.

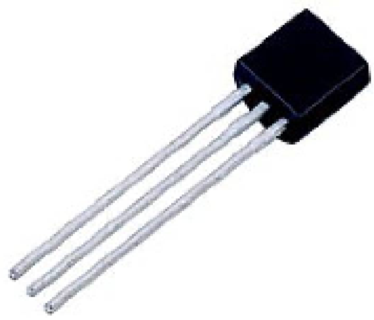


Figure 3.4 LM35 Sensor

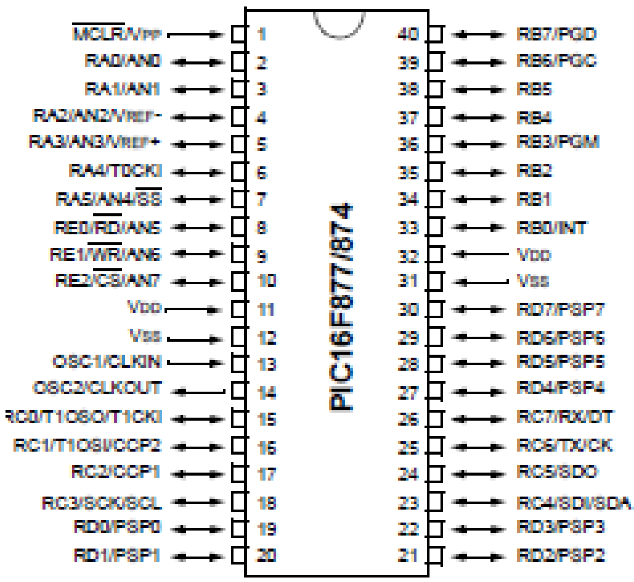


Figure 3.5 PIC16F877 Microcontroller

Figure 3.6 Cooling Fan

b. Software Design

The program was developed using the MPLAB IDE Compiler and IC-Prog. The code was formulated and compiled using the MPLAB IDE Compiler. This application is used to convert the code to machine language in which the computer can understand it. After compiling the code, it is now ready to be uploaded to the microcontroller. Using the IC-Prog application, the code was uploaded or burned to the microcontroller using also hardware that is connected to the computer and has the capability of writing the codes to the microcontroller.

The formulated code includes programs that the major components used in the system. These components are the thermal sensor, microcontroller and the cooling fan. The code is the one responsible for the automation of the whole

system. It is also responsible in the display of the system through the Liquid Crystal Display (LCD). As the temperature is measured using the thermal sensor, it serves as the input to the code uploaded in the microcontroller. The code contains various predetermined situations or functions that correspond to the measured temperature. These functions include generating the speed of the cooling fan with respect to the temperature obtained.

1. System Flowchart

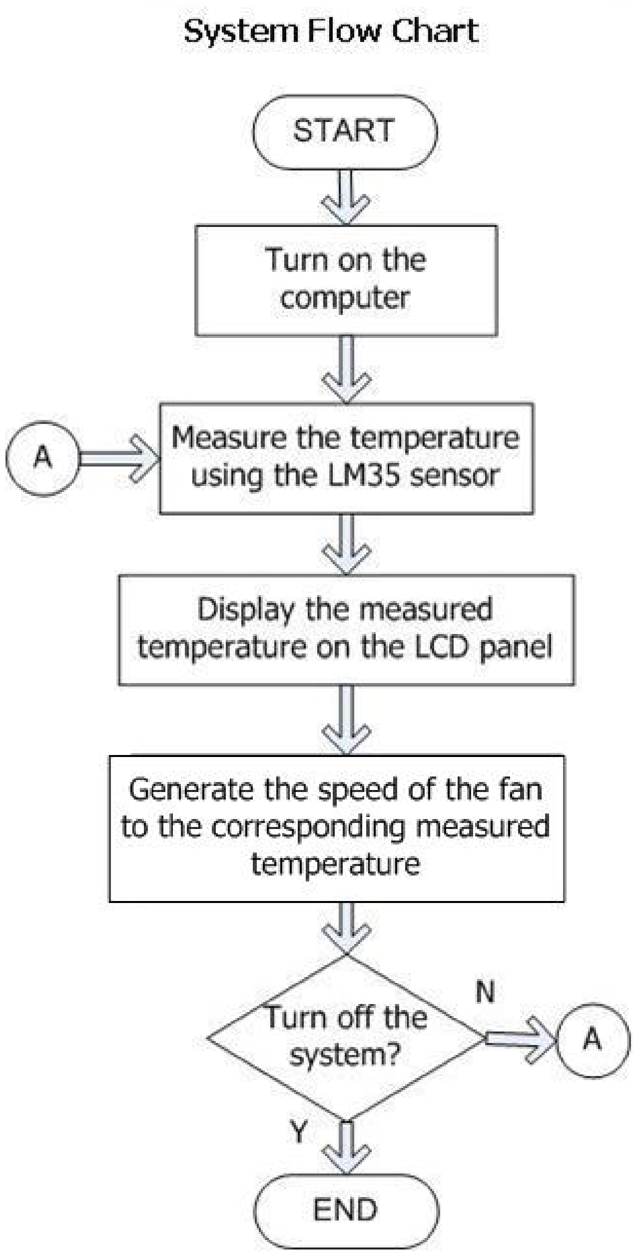


Figure 3.7 System Flowchart

The system flowchart as shown in Figure 3.7, explains how the device works internally. The computer must be turned on. After the computer is turned on, the temperature will be measured by the LM35 sensor. After measuring the

temperature, it will be displayed on the LCD panel and the corresponding speed of the fan will be generated accordingly. If the user wants to turn of the system, the process will end, if not the process will start again at measuring the temperature.

c. Prototype Development

In creating the design, the researchers gathered the necessary components and information needed in the design. Components such as hardware used were identified using the data sheets and through research. The components used were the solutions to the design problem. The following steps were used in creating the design:

1. Creation of the Circuit

   With all the information and possible solutions gathered, the researchers collected all the necessary components and designed the complete circuit of the system. The researchers used Electronics Workbench MultiSim9 as a tool in designing the circuit. The tool used is an interactive and user-friendly application that provides circuit simulator and new integrated circuits.

   The device must be first connected to the computer's power supply. Once the computer is turned on, the device will also be turned on. The LCD panel will be activated and will show the measured temperature and its corresponding fan speed. The temperature will be obtained using

the LM35 thermal sensor and it will be in degrees Celsius. While the device is running the thermal sensor will continuously measure the temperature inside the computer and this reading will serve as the input to the PIC16F877 microcontroller. The LCD panel will display the measured temperature with the corresponding speed of the fan. The PIC16F877 microcontroller is programmed to make decisions and perform functions based on predetermined situations. It will send a function to set the motor speed of the electric fan to the corresponding temperature. Lastly, the circuit will continuously be activated unless the user wants to turn of the system.

2. Creation of the Software

The Liquid Crystal Display (LCD) which serves as the display is controlled by a driver software. The microcontroller is able to store and run a program that can be programmed to perform decisions and functions based on predetermined situation. In programming the PIC microcontroller, the steps made was: formulate the code, compile the code and burn the code into a microcontroller. The MPLAB IDE compiler is used in the system. The source code will be first saved as a text file and will be run through the compiler (As shown in Figure 3.8). The compiler will read the saved text file and compiles it to its equivalent machine code (hex file). The IC-Prog is used for the hex file which is uploaded to the microcontroller (As shown in Figure 3.9). The central processing unit of

the microcontroller will be responsible in running the programmed codes. The driver software will display messages on the Liquid Crystal Display on what speed of the fan based on the measured temperature using the LM35 thermal sensor.

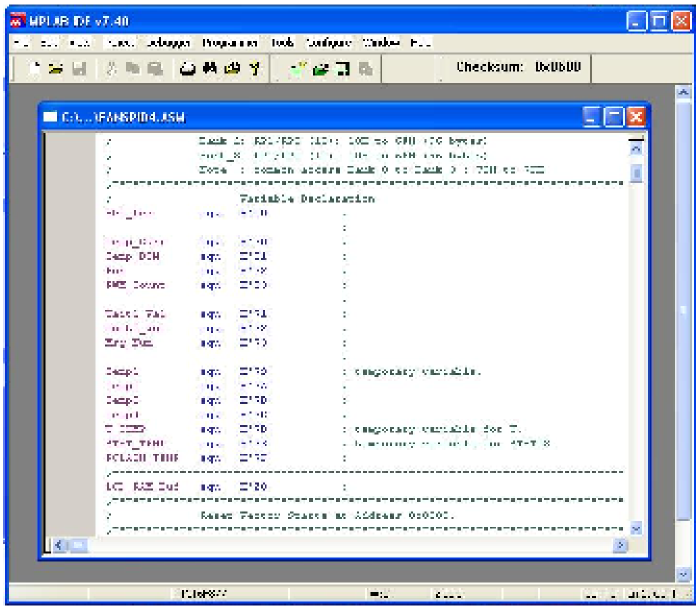


Figure 3.8 Compiling the Code using MPLAB IDE

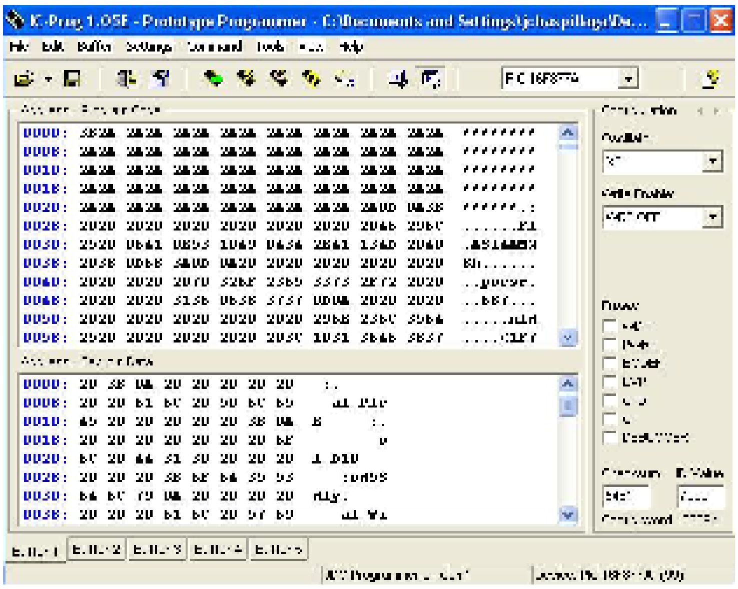


Figure 3.9 Uploading the Code to the Microcontroller Using IC-Prog

3. Integrating the Designed Circuit with the Software

The PIC16F877 microcontroller is used in the design since it has an analog input capability. The RA0 (pin 2) is used as an input for the temperature sensor (LM35), wherein the output voltage of the sensor is proportional to the temperature the sensor detects. The actual voltage of the sensor generates is 10 millivolts per degree centigrade. A capacitor is connected to the sensor as recommended by the fabricator of the LM35 to see best results.

A transistor is connected at RE2 (pin 10) which acts as a switch to supply the 12 volts motor fan. Port B, RD6 and RD7 are used for the LCD display. A LM7805 is used to regulate the voltage to 5 volts for the supply of the microcontroller and LCD display. The RA0 (pin 2) is used as an analog input for the microcontroller. The analog value generated by the sensor is converted into its digital value internally by the microcontroller. The microcontroller will send a value to RE2 which is connected to the transistor and gives a corresponding voltage depending on the value the sensor generated and then the microcontroller also send a value to the LCD to display the fan speed and the measured temperature by the sensor.

CHAPTER 4

TESTING, PRESENTATION, AND INTERPRETATION OF DATA

When the device has been completed, testing was done to determine if the device is working properly. The device must be able to measure the temperature inside the computer and it will be displayed or shown outside the computer's casing. Various tests are needed to determine the effectiveness and reliability of the device.

Testing using Multimeter

The purpose of testing the prototype using the multimeter is to prove that the fan speed increases as the voltage also increases. Using the data gathered from the test, the researchers can also calculate the power used by the device. To perform the testing of the device with a multimeter (shown on Figure 4.1 Testing the Prototype with Multimeter) the following procedures were made:

1. Place the sensor inside the digital oven.
2. Turn on the oven and adjust the temperature to the minimum limit of fan speed one (1).
3. Wait for the temperature to stabilize.
4. Tap the multimeter probes to the wires that connect the device and fan.

5. As fluctuations occur, record the read out voltage that hold the longest.
6. Redo procedures 2 to 5, adjusting the temperature to the minimum limit of each fan speed.

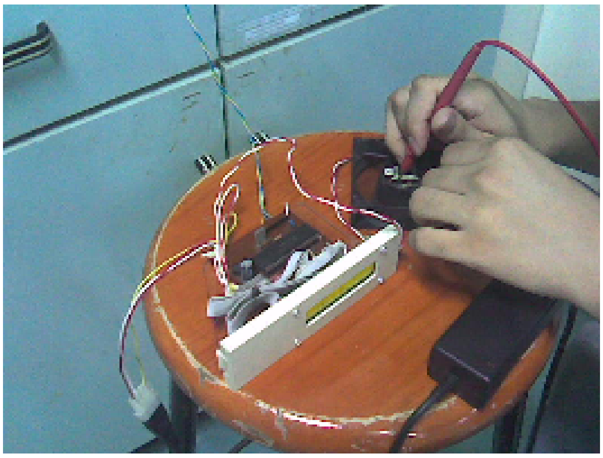


Figure 4.1 Testing Prototype with Multimeter

Table 4.1: Test of DC Fan without Device

Temperature	Voltage(DC)	Current(A)	Power(kW)
N/A	12	0.00042	5.04E-06

In Table 4.1 (Test of DC Fan without Device) as shown above, the data were obtained after testing the DC fan without the device using a multimeter. The results showed that the voltage, current and power measured is constant since the DC fan is used without the device.

Table 4.2: Test of DC Fan with Device

Temperature	Fan speed	Voltage(DC)	Power(kW)
55°C - Above	15	11.2	0.000004256
53°C - 54°C	14	10.5	0.000003675
51°C - 52°C	13	9.9	0.000003168
49°C - 50°C	12	9.3	0.00000279
47°C - 48°C	11	8.5	0.00000238
45°C - 46°C	10	7.9	0.000001975
43°C - 44°C	9	7.1	0.000001562
41°C - 42°C	8	6.5	0.0000013
39°C - 40°C	7	5.9	0.000001003
37°C - 38°C	6	5.1	0.000000765
35°C - 36°C	5	4.2	0.000000504
33°C- 34°C	4	3.4	0.00000034
31°C - 32°C	3	2.6	0.000000182
29°C - 30°C	2	2.1	0.000000105
27°C - 28°C	1	1.45	4.35E-08
26°C - Below	0	0.81	0

The results after testing the DC fan with the device with the multimeter are shown in Table 4.2 (Test of DC Fan with Device). From the data obtained, the fan speed was proven that it varies as voltage is increasing or decreasing. Also from the data table shown above, power being consumed by the device also varies when the fan speed changes. When the fan is running at higher speeds, it will consume more power and when the fan is running at lower speeds power being consumed decreases.

Testing Fan Speed with Digital Tachometer

The purpose of testing the device with a digital tachometer is to prove that each fan speed varies depending on the temperature; the researchers used a Digital Tachometer to measure the RPM for each fan speed. To perform the

testing of the device with a digital tachometer (shown on Figure 4.2 Testing Fan Speed using Tachometer) the following procedures were made:

1. Stick the reflector in the middle of the fan.
2. Place the sensor inside the digital oven.
3. Turn on the oven and adjust the temperature to the minimum limit of fan speed one (1).
4. Wait for the temperature to stabilize.
5. Position the tachometer in front of the reflector sticker.
6. As fluctuations occur, record the RPM that holds the longest.
7. Repeat procedures 3 to 6, adjusting the temperature to the minimum limit of each fan speed.

Figure 4.2 Testing Fan Speed using Tachometer

Table 4.3: Fan Speed and Corresponding Revolution per Minute

Fan Speed	Revolutions per Minute (RPM)	Temperature
15	2847	55°C - Above
14	2760	53°C - 54°C
13	2698	51°C - 52°C
12	2640	49°C - 50°C
11	2556	47°C - 48°C
10	2443	45°C - 46°C
9	2364	43°C - 44°C
8	2263	41°C - 42°C
7	2175	39°C - 40°C
6	2023	37°C - 38°C
5	1850	35°C - 36°C
4	1670	33°C- 34°C
3	1450	31°C - 32°C
2	1225	29°C - 30°C
1	803	27°C - 28°C
0	0	26°C - Below

In Table 4.3 (Fan Speed and Corresponding Revolution per Minute), as shown above, the result was that the fan speed increases as temperature increases and vice versa. From the data, it can be seen that the fan speed varies because the measured RPM (revolutions per minute) for every fan speed is varying.

Testing on Inflow Direction of Air versus Outflow Direction of Air:

The purpose of testing the inflow direction of air versus the outflow direction air is to determine which is better; inflow or outflow, as the researchers conducted a series of testing on the temperature inside the CPU. The researchers tested this for both instances wherein the device was not installed and when the

device is installed. To perform the testing of the device on which airflow is better the following procedures were made:

1. Properly place the hardware in the CPU.
2. Place the fan in an inflow direction of airflow.
3. Record the results obtained.
4. Reverse the fan in an outflow direction of airflow
5. Record the results.
6. Compare the results.
7. Repeat procedures 1-6 after installing the device.

After four hours of operation without using the device, the computer temperature was able to reach 47ºC - 48ºC. Using the device, the results to lower the temperature are the following:

Table 4.4: Inflow Direction of Airflow

Temperature (47ºC-48ºC)	Time / Duration		
	Trial 1	Trial 2	Trial 3
46ºC	1 min.	1 min. 11secs.	56 secs.
45ºC	1 min 30 secs.	1 min. 43 secs.	1 min. 50secs.
44º C	2 min. 42 secs.	2 min 26 secs.	2 min. 39 secs
43ºC	5 min. 46 secs.	6 min. 12 secs.	5 min. 53 secs.
42ºC	20 min. 5 secs.	18 min. 08 secs.	18 min. 47 secs.

Table 4.5: Outflow Direction of Airflow

Temperature (47ºC-48ºC)	Time / Duration		
	Trial 1	Trial 2	Trial 3
46ºC	44 secs.	1 min. 10 secs.	22 secs.
45ºC	1 min. 22 secs.	1 min. 55 secs.	48 secs.
44º C	2 min. 26 secs.	2 min. 26 secs.	1 min 22 secs.
43ºC	4 min. 06 secs.	4 min. 40 secs.	2 min. 49 secs.
42ºC	11 min. 32 secs.	18 min. 22 secs.	16 min. 13 secs.

From Table 4.4 (Inflow Direction of Airflow) and Table 4.5 (Outflow Direction of Airflow) it shows the time it takes to lower the temperature and also the time it takes until the temperature is stable from the highest temperature obtained. Also from the data gathered, the result was that the device was able to reduce the temperature inside the computer from the highest reading which is 48ºC down to 42ºC.

The computer was turned off for four hours for the temperature to go down. The device was installed first before the computer was turned on and the results for temperature reading are the following: (The initial temperature is 32ºC)

Table 4.6: Outflow Direction of Airflow since Start-Up

Temperature (32-33ºC)	Time / Duration		
	Trial 1	Trial 2	Trial 3
34ºC	44 secs.	32 secs.	58 secs.
35º C	52 secs.	1 min. 12 secs.	1 min. 27 secs.
36ºC	2 min. 32 secs.	2 min. 52 secs.	2 min. 48 secs.
37ºC	3 min. 40 secs.	3 min. 27 secs.	3 min. 33 secs.
38 ºC	3 min. 32 secs.	4 min. 17 secs.	3 min. 59 secs.
39 ºC	4 min. 43 secs.	5 min. 33 secs.	5 min. 41 secs.

From the data in Table 4.6 (Outflow Direction of Airflow), the results were the temperature range was 32ºC - 39ºC. The computer was turned on and was operating for four hours and the temperature has reached only at a maximum of 39ºC.

Table 4.7: Inflow Direction of Airflow since Start-Up

Temperature (32-33ºC)	Time / Duration		
	Trial 1	Trial 2	Trial 3
34ºC	33 secs.	26 secs.	42 secs.
35º C	56 secs.	1 min. 22 secs.	1 min. 12 secs.
36ºC	2 min. 12 secs.	2 min. 37 secs.	2 min. 22 secs.
37ºC	3 min. 35 secs.	3 min. 12 secs.	3 min. 52 secs.
38 ºC	3 min. 49 secs.	4 min. 15 secs.	4 min. 8 secs.
39 ºC	4 min. 37 secs.	5 min. 24 secs.	4 min. 56 secs.
40ºC	6 min. 09 secs.	7 min. 13 secs.	6 min. 43 secs.
41ºC	3 min. 32 secs.	4 min. 18 secs.	3 min. 42 secs.

From the data in Table 4.7 (Inflow Direction of Airflow) as shown above, it was observed that when using the device since startup time, the temperature

range is 32-41ºC. The computer unit was operating for a total of four hours and the highest temperature it obtained was 41ºC. The temperature inside the computer was maintained at a constant temperature and did not increased again.

After performing all the tests and gathering all the data needed to compare on what airflow is better, it can be assumed that outflow direction of air is better. These can be seen on the results that showed that from the highest temperature, it takes lesser time to lower it comparing on the results when the direction of air is inflow. Also from the data obtained, when the direction of air is outflow or exhaust the temperature inside the computer is maintained at a lower temperature (39ºC) compared to when the direction of air is inflow which is (41ºC).

Testing on a VOIP server

The device was tested on actual application for Voice over Internet Protocol server. The prototype was tested in 48 hours uptime of the server. This purpose of performing the test is to show how the prototype helps the server in maintaining desirable temperature in the CPU. To perform the testing of the device on a VOIP server (shown on Figure 4.3 Softphones used for testing in Voice over IP server) the following procedures were made:

1. Properly install the prototype on the server.
2. Make sure that the computers are properly connected with the server.

3. Soft phones or IP phones should be properly installed in the computer.
4. Turn-on the server for 48 hours.
5. Make random number of phone calls in between hours of testing.
6. Record the data gathered.

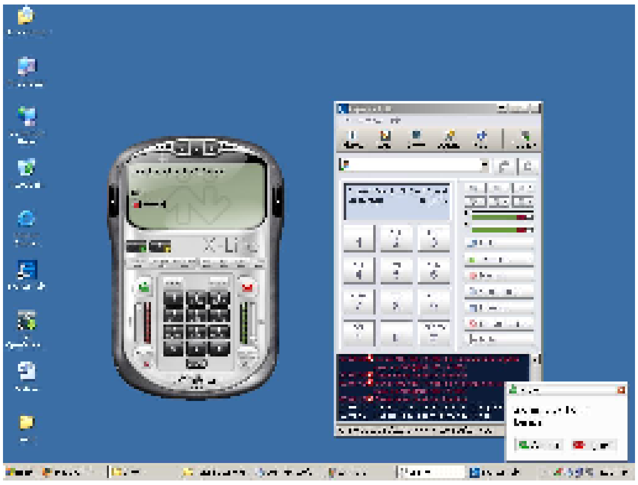


Figure 4.3 Softphones used for testing in Voice over IP server

Table 4.8: Test on the first few minutes after boot-up at initial CPU temperature of 29ºC

Room Temperature	Temperature	Duration
25 ºC	30	11 sec
25 ºC	31	46 sec
25 ºC	32	1 min 25 sec
25 ºC	33	1 min 56 sec
25 ºC	34	2 min 42 sec
25 ºC	35	2 min 56 sec
25 ºC	36	4 min 24 sec
25 ºC	37	5 min 10 sec
25 ºC	36	6 min 16 sec
25 ºC	35	6 min 4 sec
25 ºC	36	7 min 8 sec
25 ºC	37	7 min 16 sec
25 ºC	38	15 min 27 sec
25 ºC	37	17 min 42 sec
25 ºC	36	18 min 51 sec
25 ºC	35	22 min 17 sec

From the data on Table 4.8 (Test on the first few minutes after boot-up at initial CPU temperature of 29ºC) it was shown that on the first few minutes after booting up, the temperature of the computer was not stable. The temperature was continuously increasing and decreasing especially on the first seven minutes of the computer after boot-up, then slowly changing on the next few minutes until it reaches a steady temperature.

Table 4.9: Test on the Server after Reaching Steady State

Hours Duration	Lowest Temperature	Highest Temperature	Number of phone calls
1	29	38	34 calls
2	35	35	32 calls
3	35	35	40 calls
4	35	35	35 calls
5-7	35	35	46 average calls
8	35	36	42 calls
9-16	35	35	55 average calls
17	34	35	50 calls
18-24	34	34	36 average calls
25	34	35	35 calls
26-34	35	35	26 average calls
35	35	36	41 calls
36-44	35	35	34 average calls
45	34	35	37 calls
46-48	35	35	40 average calls

The room temperature was at 25 degree Celsius**.** After performing the test on the server as shown in Table 4.9 (Test on the Server after Reaching Steady State), it showed that after the server reaches its steady temperature. The server's temperature was maintained to 35 degree Celsius. The temperature changes for a while after few hours but will still return to 35 degree Celsius. The number of phone calls did not have an effect on the changing of server's temperature.

Chapter 5

CONCLUSION AND RECOMMENDATION

Conclusion

The researchers were able to develop a new hardware called Microcontroller-Based PC Thermometer with Regulated Fan. When the system was completed, the researchers gained new knowledge on how to use a microcontroller, a thermal sensor and how to integrate these components together. The PC thermometer with regulated fan was able to display the temperature inside the CPU using the thermal sensor and LCD display. The speed of the cooling fan was proven that it varies depending on the measured temperature through the use of digital tachometer. Through the testing, it showed that the temperature inside the body of the computer was maintained to 34-36 degrees Celsius when the computer is running at 48 hours at room temperature of 25 degrees Celsius. The hardware can be a big help for monitoring and maintaining proper temperature inside the CPU.

Recommendation

Several improvements can be applied to the device to further enhance its capabilities in monitoring the server to avoid downtime. Humidity control system can be added in the prototype to maintain proper humidity in the server room. It can be a big help to avoid moisture whenever it is too humid in the server room.

Power sensor can also be added before the UPS. It can detect and report on the availability and loss of AC power. It will be a big help to the area that typically face power outages, and if the administrator want to be alerted before the UPS goes down too.

REFERENCES

Christensen, Jim, *Electronics Design, Strategy, News*. March 21, 2002.

Denmark, Bruce, *Electronics Design, Strategy, News*. September 28, 2000.

Guy, John, *Electronics Design, Strategy, News*. March 4, 2004.

Predro, Myke, *Programming and Customizing PIC Micro Microcontrollers Second Edition*. April 2000.

Predro, Myke, *Handbook of Microcontrollers*. December 1997.

Gay, Ken, *Electronics Design*. August 29, 2007.

Vallis, Darrin, *Electronics Design.* 2004.

APPENDIX A

Source Code

```
;*****************************
******************************
*****************

;		File FANSPID4.ASM @ 8Mhz
		processor	16F877
		include <P16F877.inc>
	__config  _XT_OSC & _WDT_OFF & _PWRTE_ON & _LVP_OFF & _BODEN_OFF & _CP_ALL
;*****************************
******************************
*****************
;		General Purpose RAM location: (STATUS-reg RP1/RP0: x__x xxxx)
;		Bank_0: RP1/RP0 (00): 20H to 7FH (96 bytes)
;		Bank_1: RP1/RP0 (01): 20H to 6FH (80 bytes)
;		Bank_2: RP1/RP0 (10): 10H to 6FH (96 bytes)
;		Bank_3: RP1/RP0 (11): 10H to 6FH (96 bytes)
;		Note  : common access Bank_0 to Bank_3 : 70H to 7FH
;*****************************
******************************
*****************
;			Variable Declaration
ADC_Reg		equ	H'20'		;
							;
Temp_Curr	equ	H'30'		;
Temp_BIN	equ	H'31'		;
Fan		equ	H'32'		;
PWM_Count	equ	H'33'		;
							;
Wait1_Val	equ	H'71'		;
Wait2_Val	equ	H'72'		;
Msg_Num		equ	H'73'		;
							;
Temp1		equ	H'79'		; temporary variable.
Temp2		equ	H'7A'		;
Temp3		equ	H'7B'		;
Temp4		equ	H'7C'		;
W_TEMP		equ	H'7D'		; temporary variable for W.
STAT_TEMP	equ	H'7E'		; temporary variable for STATUS.
PCLATH_TEMP	equ	H'7F'		;
;-------------------------------------------
--------------------------------
LCD_RAM_Buf	equ	H'20'		;
;*****************************
******************************
*****************
;		Reset Vector Starts at Address 0x0000.
```

```
;*****************************************************************************

        org   0x0000         ; start of reset vector.
        goto  Initialize     ;
                             ;
        org   0x0004         ; start of interrupt service routine.
        goto  ISR_routine    ;
;*****************************************************************************
;       Initialization Routine.
;*****************************************************************************
Initialize:   clrf  TMR0           ; Clear TMR0
        clrf  INTCON         ; Disable Interrupts and clear T0IF
        bcf   STATUS,RP1     ;
        bsf   STATUS,RP0     ; Select Bank 1
        movlw B'11000100'    ; prescaler of 1:32
        movwf OPTION_REG     ;
                             ;
        movlw B'00001110'    ;Set AN0 as  Analog input
        movwf ADCON1         ;
        movlw B'11011111'    ; 0=OUT 1=IN
        movwf TRISA          ; Port A. 11xx xxxx:TTL
                             ;
        movlw B'00000000'    ; 0=OUT 1=IN
        movwf TRISB          ; Port B. xxxx xxxx:TTL
                             ;
        movlw B'00000000'    ; 0=OUT 1=IN
        movwf TRISC          ; Port C. xxxx xxxx:schmitt
                             ;
        movlw B'00000000'    ; 0=OUT 1=IN
        movwf TRISD          ; Port D. xxxx xxxx:schmitt
                             ;
        movlw B'00000000'    ; 0=OUT 1=IN
        movwf TRISE          ; Port E. xxxx xxxx:schmitt
                             ;
        bcf   STATUS,RP0     ; Select Bank 0
                             ;
        call  Init_Var       ;
        call  Init_ADC       ;
        call  Init_LCD       ;
```

```
        call  Disp_LCD        ;
                              ;
        bsf   INTCON,T0IE     ; Enable TMR0 Interrupt.
        bsf   INTCON,GIE      ; Enable All Interrupts.
;****************************************************************************
;       Main Program Starts Here.
;****************************************************************************
Main:   nop                   ;
        goto  Main            ;
;****************************************************************************
;       The Interrupt Service Routine.
;****************************************************************************
ISR_routine:                  ; Save Registers
        movwf W_TEMP          ; W -> W_TEMP
        movf  STATUS,W        ; STATUS -> W
        movwf STAT_TEMP       ; W -> STAT_TEMP
        bcf   STATUS,RP0      ; Bank 0
                              ; Check which interrupt has occurred.
        btfsc INTCON,T0IF     ; Timer0 Interrupt ?
        goto  TMR0int         ;
                              ; Other causes, disregard!
RestoreReg:                   ; Restore Registers
        movf  STAT_TEMP,W     ; STAT_TEMP -> W
        movwf STATUS          ; W -> STATUS
        movf  W_TEMP,W        ; W_TEMP -> W
                              ;
        retfie                ; Return from Interrupt.
;****************************************************************************
;       TIMER 0 (TMR0) Interrupt Service Routine.
;****************************************************************************
TMR0int: bcf  INTCON,T0IF     ; Reset TMR0 Overflow Flag.
        movlw D'06'           ; store value to TMR0
        addwf TMR0,F          ;
                              ;
        call  Read_ADC        ;
```

```
            call  Do_PWM          ;

            call  Disp_Data       ;

            call  Disp_LCD        ;

                                  ;

TMR0intX:     goto  RestoreReg      ;
done! Restore registers & exit.

;****************************************************************************

Msg0:       addwf PCL,F             ;

             ;0123456789012345

            dt   "  Temp: __ degC "

            dt   "  Fan : __      "


Init_Var:     clrf  Msg_Num           ;

            call  Ld_Msg2RAM        ;

            clrf  Temp_Curr         ;

            clrf  Temp_BIN          ;

            clrf  PORTA             ;

            clrf  PORTE             ;

            clrf  PORTC             ;

            clrf  PORTD             ;

            return                  ;

;****************************************************************************

;                   Variable Declaration

;****************************************************************************

ADC0_HI       equu   ADC_Reg
;

ADC0_LO       equ   ADC_Reg +1
;

ADC_Sel       equ   ADC_Reg +4
;

ADC_DataH     equ   ADC_Reg +5
;

ADC_DataL     equ   ADC_Reg +6
;

ADC_Dly       equ   ADC_Reg +7
;

                                ;

Curr10        equ   ADC_Reg +9
;

Curr01        equ   ADC_Reg +D'10'
;

                                ;

Count         equ   ADC_Reg +D'11'
;

Unit          equ   ADC_Reg +D'12'
;

Ten           equ   ADC_Reg +D'13'
;

Hundred       equ   ADC_Reg +D'14'
;

;****************************************************************************

Init_ADC:     clrf   Curr10          ;
```

```
        clrf   Curr01          ;
        clrf   Count           ;
                               ;
        movlw  B'00000001'     ; 00xx x001
        movwf  ADCON0          ; select AN0 to convert
        clrf   ADC_Sel         ;
        clrf   ADC_DataH       ;
        clrf   ADC_DataL       ;
        clrf   ADC0_HI         ;
        clrf   ADC0_LO         ;
        clrf   ADC_Dly         ;
                               ;
        return                 ;
;*************************************************************************
Read_ADC:    incf   ADC_Dly,F  ;
        movlw  D'30'           ;
        subwf  ADC_Dly,W       ;
        btfsc  STATUS,C        ;
        clrf   ADC_Dly         ;
                               ;
        movf   ADC_Dly,W       ;
        sublw  D'0'            ;
        btfss  STATUS,Z        ;
        goto   Read_ADCX       ;
                               ;
        movf   ADRESH,W        ; get A/D result
        movwf  ADC_DataH       ; save to ADC_Data
        movlw  ADRESH          ;
        movwf  FSR             ;
        bsf    FSR,7           ;
        movf   INDF,W          ;
        movwf  ADC_DataL       ;
        movlw  B'00000001'     ;
        movwf  ADCON0          ; ensure A/D is active
                               ;
        movf   ADC_DataH,W     ;
        movwf  ADC0_HI         ;
        movf   ADC_DataL,W     ;
        movwf  ADC0_LO         ;
        movlw  B'00000001'     ; 00xx x001
        movwf  ADCON0          ; select AN1 to convert
        call   Get_Temp        ;
        call   Get_Fan         ;
                               ;
        clrf   ADC_Sel         ;
                               ;
```

```
Read_ADCX:    movf    ADC_Dly,W
;
              sublw   D'25'            ;
              btfsc   STATUS,Z         ;
              bsf     ADCON0,2         ;
start A/D conversion
              return                   ;
;*****************************
*****************************
****************
Get_Temp:     movf    ADC0_HI,W
;
              movwf   Temp2            ;
              movf    ADC0_LO,W        ;
              movwf   Temp1            ;
              bcf     STATUS,C         ;
              rlf     Temp1,F          ;
              rlf     Temp2,F          ;
              movf    Temp2,W          ;
              movwf   Temp_BIN         ;
              call    BIN2BCD          ;
              movf    Ten,W            ;
              swapf   Ten,W            ;
              iorwf   Unit,W           ;
              movwf   Temp_Curr        ;
Get_TempX:    return                   ;
;*****************************
*****************************
****************
BIN2BCD:      clrf    Unit             ;
              clrf    Ten              ;
              clrf    Hundred          ;
                                       ;
Dec100Lp:     movlw   D'100'
;
              subwf   Temp2,W          ;
              btfss   STATUS,C         ;
              goto    Dec10Lp
;Temp2 <  100
              movwf   Temp2
;Temp2 >= 100
              incf    Hundred,F        ;
              goto    Dec100Lp         ;
                                       ;
Dec10Lp:      movlw   D'10'            ;
              subwf   Temp2,W          ;
              btfss   STATUS,C         ;
              goto    Dec1Lp           ;
              movwf   Temp2            ;
              incf    Ten,F            ;
              movlw   D'10'            ;
              goto    Dec10Lp          ;
                                       ;
Dec1Lp:       movf    Temp2,W
;
              movwf   Unit             ;
              return                   ;
```

```
;*****************************************************************************
Get_Fan:    clrf    Fan             ;
            movlw   D'25'           ;
            subwf   Temp_BIN,W      ;
            btfss   STATUS,C        ;
            goto    Get_Fan_A       ;
            movwf   Fan             ;
Get_Fan_A:  bcf     STATUS,C
;
            rrf     Fan,F           ;
            movlw   D'15'           ;
            subwf   Fan,W           ;
            btfss   STATUS,C        ;
            goto    Get_FanX        ;
            movlw   D'15'           ;
            movwf   Fan             ;
Get_FanX:   return                  ;
;*****************************************************************************
Do_PWM:     movf
PWM_Count,W     ;
            subwf   Fan,W           ;
            btfss   STATUS,C        ;
            goto    PWM_OFF         ;
PWM_ON:     bsf     PORTE,2
;
            goto    Inc_PWM         ;
PWM_OFF:    bcf     PORTE,2
;
                                    ;
Inc_PWM:    incf    PWM_Count,F
;
            movlw   D'16'           ;
            subwf   PWM_Count,W
;
            btfss   STATUS,C        ;
            goto    Do_PWMX         ;
            clrf    PWM_Count       ;
                                    ;
Do_PWMX:    return                  ;
;*****************************************************************************
Disp_Data:  movlw  LCD_RAM_Buf
;
            addlw  D'8'             ;
            movwf  FSR              ;
            bsf    FSR,7            ;
            swapf  Temp_Curr,W      ;
            andlw  H'0F'            ;
            addlw  H'30'            ;
            movwf  INDF             ;
            incf   FSR,F            ;
            movf   Temp_Curr,W      ;
            andlw  H'0F'            ;
```

```
            addlw  H'30'           ;
            movwf  INDF            ;
                                   ;
            movf   Fan,W           ;
            movwf  Temp2           ;
            call   BIN2BCD         ;
            movlw  LCD_RAM_Buf
;
            addlw  D'24'           ;
            movwf  FSR             ;
            bsf    FSR,7           ;
            movf   Ten,W           ;
            andlw  H'0F'           ;
            addlw  H'30'           ;
            movwf  INDF            ;
            incf   FSR,F           ;
            movf   Unit,W          ;
            andlw  H'0F'           ;
            addlw  H'30'           ;
            movwf  INDF            ;
Disp_DataX:   return               ;
;*****************************************************************************
;           LCD Subroutine         ;
;*****************************************************************************
;VARIABLE USED                     ;
;Wait1_Val                         ;
;Wait2_Val                         ;
;Msg_Num                           ;
;Temp1                             ;
;Temp2                             ;
;Temp3                             ;
;Temp4                             ;
                                   ;
LCD_DPort     equu    PORTB
;
LCD_CPort     equ    PORTD
;
LCD_EN        equ    7             ;
LCD_RS        equ    6             ;
                                   ;
LCD_Line_Max  equ    D'2'          ;
LCD_Char_Max  equ    D'16'
;
LCD_L1_Addr   equ    D'00' +H'80'
; = H'80'
LCD_L2_Addr   equ    LCD_L1_Addr
+D'40' ; = H'A8'
                                   ;
Set_RS0:      macro                ;
        bcf    LCD_CPort,LCD_RS    ;
        endm                       ;
                                   ;
```

```
Set_RS1:      macro                 ;
              bsf    LCD_CPort,LCD_RS  ;
              endm                  ;
                                   ;
Pulse_EN:     bsf
LCD_CPort,LCD_EN  ;
              nop                   ;
              nop                   ;
              nop                   ;
              nop                   ;
              bcf    LCD_CPort,LCD_EN
;
              call   Wait1          ;
              return                ;
                                   ;
;*****************************************************************************
Init_LCD:     Set_RS0
;set RS to 0
              movlw  D'200'
;load 10mS delay
              call   Wait2          ;
              movlw  D'200'
;load 10mS delay
              call   Wait2          ;
                                   ;
              movlw  H'38'          ;set
LCD to 8 Bit Data, 2 line display
              movwf  LCD_DPort      ;
              call   Pulse_EN       ;
              movlw  D'100'
;load 5mS delay
              call   Wait2          ;
                                   ;
              call   Pulse_EN       ;
              movlw  D'100'
;load 5mS delay
              call   Wait2          ;
                                   ;
              call   Pulse_EN       ;
              movlw  D'100'
;load 5mS delay
              call   Wait2          ;
                                   ;
              movlw  H'06'          ;set
Display Increment, No Shift
              movwf  LCD_DPort      ;
              call   Pulse_EN       ;
                                   ;
              movlw  H'0F'          ;set
Disp=ON, Cursor=ON, Blink=ON
              movlw  H'0C'          ;set
Disp=ON, Cursor=OFF, Blink=OFF
              movwf  LCD_DPort      ;
              call   Pulse_EN       ;
                                   ;
              movlw  H'14'          ;set
CursorMove, Shift to Right
```

```
        movwf  LCD_DPort        ;
        call   Pulse_EN         ;
                                ;
        movlw  H'01'
;Clear LCD Display
        movwf  LCD_DPort        ;
        call   Pulse_EN         ;
                                ;
        movlw  D'100'
;load 5mS delay
        call   Wait2            ;
                                ;
        return                  ;
                                ;
Wait1:      movlw  H'10'
;approx. 50uS delay
        movwf  Wait1_Val        ;
Wait1_loop: decf   Wait1_Val,F
;
        btfss  STATUS,Z         ;
        goto   Wait1_loop       ;
        return                  ;
                                ;
Wait2:      movwf  Wait2_Val
;N x 50uS delay
Wait2_loop  call   Wait1        ;
        decf   Wait2_Val,F      ;
        btfss  STATUS,Z         ;
        goto   Wait2_loop       ;
        return                  ;
;***************************
****************************
*****************
Disp_LCD:                       ;
                                ;
Disp_LCD1:  Set_RS0             ;
        movlw  LCD_L1_Addr      ;
        movwf  LCD_DPort        ;
        call   Pulse_EN         ;
        Set_RS1                 ;
        clrf   Temp1            ;
RAM2LCD1:   movlw
LCD_Char_Max    ;
        subwf  Temp1,W          ;
        btfsc  STATUS,Z         ;
        goto   RAM2LCD1X        ;
        movlw  LCD_RAM_Buf
;
        addwf  Temp1,W          ;
        movwf  FSR              ;
        bsf    FSR,7
;Ind_Addr Select Bank 1
        movf   INDF,W           ;
        movwf  LCD_DPort        ;
        call   Pulse_EN         ;
        incf   Temp1,F          ;
```

```
            goto   RAM2LCD1          ;
RAM2LCD1X:    nop                    ;
                                     ;
Disp_LCD2:    Set_RS0                ;
            movlw  LCD_L2_Addr       ;
            movwf  LCD_DPort         ;
            call   Pulse_EN          ;
            Set_RS1                  ;
            clrf   Temp1             ;
RAM2LCD2:     movlw
LCD_Char_Max      ;
            subwf  Temp1,W           ;
            btfsc  STATUS,Z          ;
            goto   RAM2LCD2X         ;
            movlw  LCD_RAM_Buf
;
            addlw  LCD_Char_Max
;
            addwf  Temp1,W           ;
            movwf  FSR               ;
            bsf    FSR,7
;Ind_Addr Select Bank 1
            movf   INDF,W            ;
            movwf  LCD_DPort         ;
            call   Pulse_EN          ;
            incf   Temp1,F           ;
            goto   RAM2LCD2          ;
RAM2LCD2X:    nop                    ;
                                     ;
            return                   ;
;****************************
*****************************
****************
Ld_Msg2RAM:   clrf   Temp1
;
            clrf   Temp3             ;
            movf   Msg_Num,W         ;
            movwf  Temp1             ;
                                     ;
Ld_Msg_Adr:   movf   Temp1,W
;
            btfsc  STATUS,Z          ;
            goto   Ld_MsgLoop        ;
            movlw  D'32'             ;
            addwf  Temp3,F           ;
            decf   Temp1,F           ;
            goto   Ld_Msg_Adr        ;
                                     ;
Ld_MsgLoop:   movlw  D'32'
;
            subwf  Temp1,W           ;
            btfsc  STATUS,Z          ;
            goto   Ld_MsgDone        ;
                                     ;
            movf   PCLATH,W          ;
            movwf  Temp4             ;
```

```
            movlw  HIGH Msg0         ;
            movwf  PCLATH            ;
            movf   Temp1,W           ;
            addwf  Temp3,W           ;
            call   Msg0              ;
            movwf  Temp2             ;
            movf   Temp4,W           ;
            movwf  PCLATH            ;
            goto   Ld_Msg_Char       ;
                                 ;
Ld_Msg_Char:  movlw
LCD_RAM_Buf       ;
            addwf  Temp1,W           ;
            movwf  FSR               ;
            bsf    FSR,7
;Ind_Addr Select Bank 1
            movf   Temp2,W           ;
            movwf  INDF              ;
            incf   Temp1,F           ;
            goto   Ld_MsgLoop        ;
Ld_MsgDone:   return                 ;
;****************************
*****************************
****************
            end                  ;
;****************************
*****************************
***************
```

APPENDIX B

PIC16F87 Data Sheet

PIC16F87X

28/40-Pin 8-Bit CMOS FLASH Microcontrollers

Devices Included in this Data Sheet:

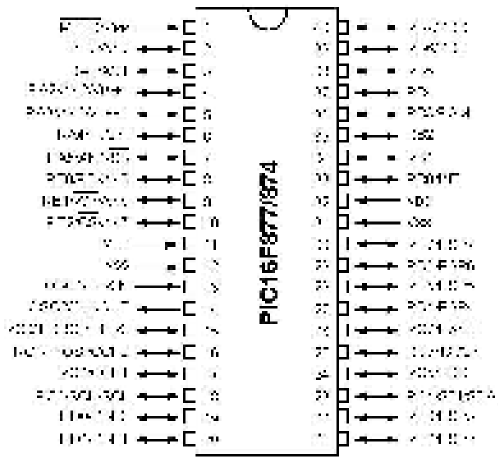
- PIC16F873
- PIC16F874
- PIC16F876
- PIC16F877

Microcontroller Core Features:

- High performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM)
  Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to the PIC16C73B/74B/76/77
- Interrupt capability (up to 14 sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and
  Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low power, high speed CMOS FLASH/EEPROM technology
- Fully static design
- In-Circuit Serial Programming™ (ICSP) via two pins
- Single 5V In-Circuit Serial Programming capability
- In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial, Industrial and Extended temperature ranges
- Low-power consumption:
  - < 0.6 mA typical @ 3V, 4 MHz
  - 20 µA typical @ 3V, 32 kHz
  - < 1 µA typical standby current

Pin Diagram

PDIP



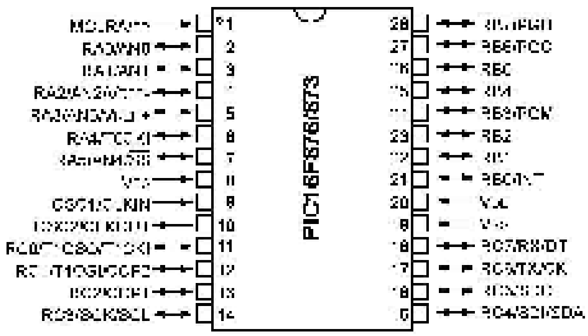
Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- 10-bit multi-channel Analog-to-Digital converter
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I²C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

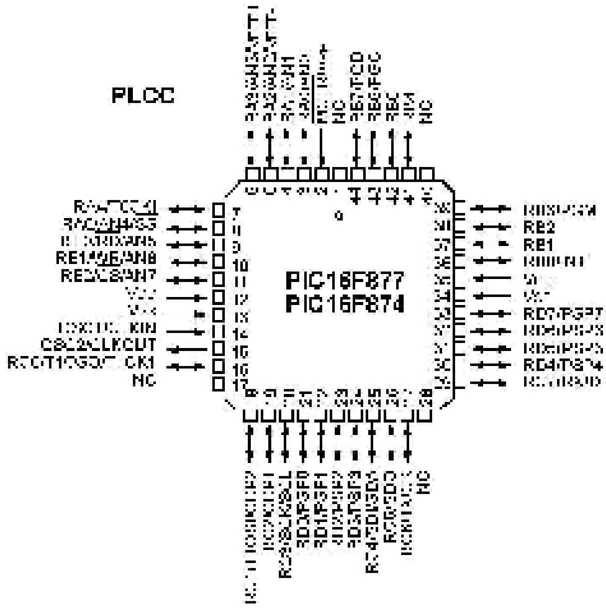
© 2001 Microchip Technology Inc.

Pin Diagrams

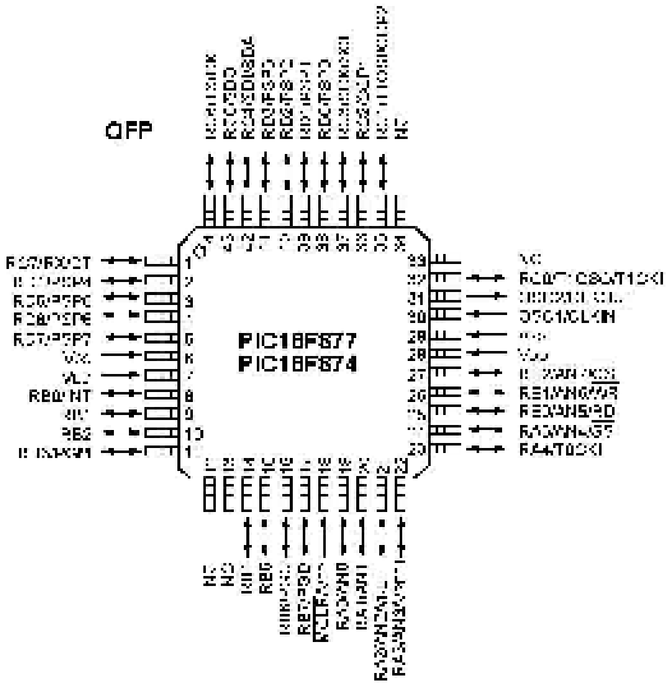
PDIP, SOIC



PLCC



QFP



© 2001 Microchip Technology Inc.

Key Features PICmicro™ Mid-Range Reference Manual (DS33023)	PIC16F873	PIC16F874	PIC16F876	PIC16F877
Operating Frequency	DC - 20 MHz	DC - 20 MHz	DC - 20 MHz	DC - 20 MHz
RESETS (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
FLASH Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory	128	128	256	256
Interrupts	13	14	13	14
I/O Ports	Ports A,B,C	Ports A,B,C,D,E	Ports A,B,C	Ports A,B,C,D,E
Timers	3	3	3	3
Capture/Compare/PWM Modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Instruction Set	35 instructions	35 instructions	35 instructions	35 instructions

1.0 DEVICE OVERVIEW

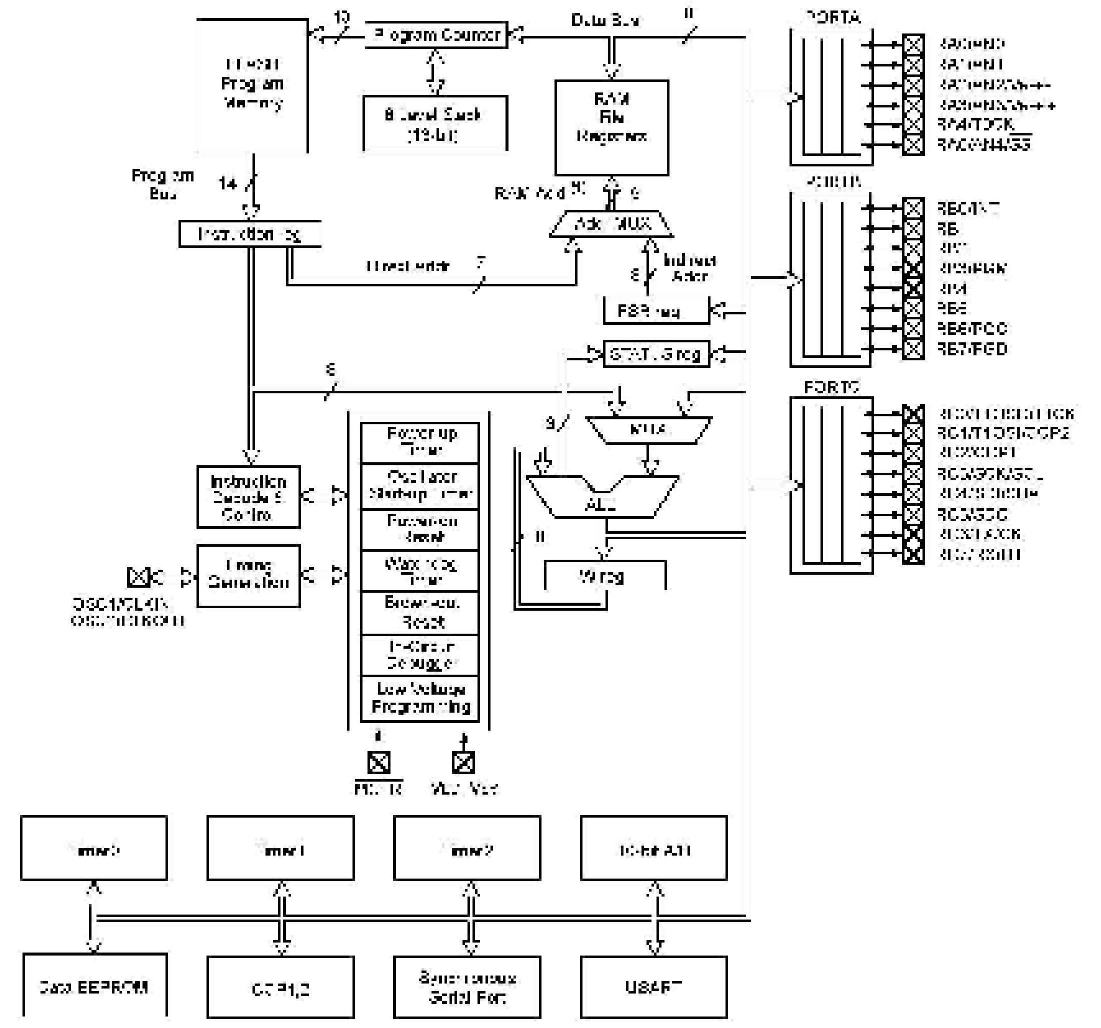
This document contains device specific information. Additional information may be found in the PICmicro™ Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip website. The Reference Manual should be considered a complementary document to this data sheet, and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

There are four devices (PIC16F873, PIC16F874, PIC16F876 and PIC16F877) covered by this data sheet. The PIC16F876/873 devices come in 28-pin packages and the PIC16F877/874 devices come in 40-pin packages. The Parallel Slave Port is not implemented on the 28-pin devices.

The following device block diagrams are sorted by pin number; 28-pin for Figure 1-1 and 40-pin for Figure 1-2. The 28-pin and 40-pin pinouts are listed in Table 1-1 and Table 1-2, respectively.

FIGURE 1-1: PIC16F873 AND PIC16F876 BLOCK DIAGRAM

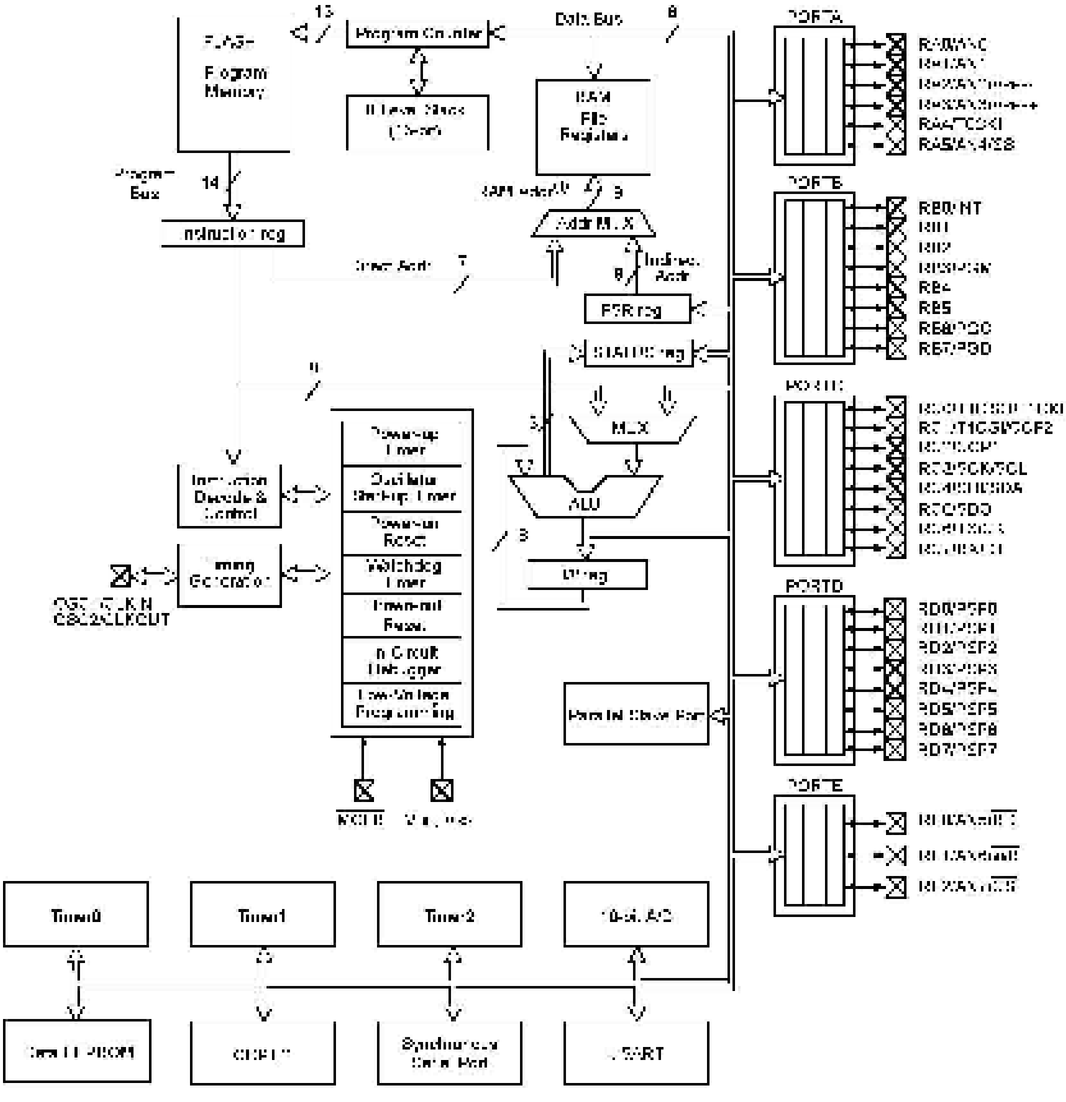
Device	Program FLASH	Data Memory	Data EEPROM
PIC16F873	4K	192 Bytes	128 Bytes
PIC16F876	8K	368 Bytes	256 Bytes



Note 1: Higher order bits are from the STATUS register.

FIGURE 1-2: PIC16F874 AND PIC16F877 BLOCK DIAGRAM

Device	Program FLASH	Data Memory	Data EEPROM
PIC16F874	4K	192 Bytes	128 Bytes
PIC16F877	8K	368 Bytes	256 Bytes



Note 1: Higher order bits are from the STATUS register.

© 2001 Microchip Technology Inc.

TABLE 1-1: PIC16F873 AND PIC16F876 PINOUT DESCRIPTION

Pin Name	DIP Pin#	SOIC Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	9	9	I	ST/CMOS[3]	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	10	10	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, the OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
$\overline{MCLR}/V_{PP}$	1	1	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device.
					PORTA is a bi-directional I/O port.
RA0/AN0	2	2	I/O	TTL	RA0 can also be analog input0.
RA1/AN1	3	3	I/O	TTL	RA1 can also be analog input1.
RA2/AN2/VREF-	4	4	I/O	TTL	RA2 can also be analog input2 or negative analog reference voltage.
RA3/AN3/VREF+	5	5	I/O	TTL	RA3 can also be analog input3 or positive analog reference voltage.
RA4/T0CKI	6	6	I/O	ST	RA4 can also be the clock input to the Timer0 module. Output is open drain type.
RA5/$\overline{SS}$/AN4	7	7	I/O	TTL	RA5 can also be analog input4 or the slave select for the synchronous serial port.
					PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	21	21	I/O	TTL/ST[1]	RB0 can also be the external interrupt pin.
RB1	22	22	I/O	TTL	
RB2	23	23	I/O	TTL	
RB3/PGM	24	24	I/O	TTL	RB3 can also be the low voltage programming input.
RB4	25	25	I/O	TTL	Interrupt-on-change pin.
RB5	26	26	I/O	TTL	Interrupt-on-change pin.
RB6/PGC	27	27	I/O	TTL/ST[2]	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock.
RB7/PGD	28	28	I/O	TTL/ST[2]	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data.
					PORTC is a bi-directional I/O port.
RC0/T1OSO/T1CKI	11	11	I/O	ST	RC0 can also be the Timer1 oscillator output or Timer1 clock input.
RC1/T1OSI/CCP2	12	12	I/O	ST	RC1 can also be the Timer1 oscillator input or Capture2 input/Compare2 output/PWM2 output.
RC2/CCP1	13	13	I/O	ST	RC2 can also be the Capture1 input/Compare1 output/PWM1 output.
RC3/SCK/SCL	14	14	I/O	ST	RC3 can also be the synchronous serial clock input/output for both SPI and I²C modes.
RC4/SDI/SDA	15	15	I/O	ST	RC4 can also be the SPI Data In (SPI mode) or data I/O (I²C mode).
RC5/SDO	16	16	I/O	ST	RC5 can also be the SPI Data Out (SPI mode).
RC6/TX/CK	17	17	I/O	ST	RC6 can also be the USART Asynchronous Transmit or Synchronous Clock.
RC7/RX/DT	18	18	I/O	ST	RC7 can also be the USART Asynchronous Receive or Synchronous Data.
V_{SS}	8, 19	8, 19	P	—	Ground reference for logic and I/O pins.
V_{DD}	20	20	P	—	Positive supply for logic and I/O pins.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

© 2001 Microchip Technology Inc.

TABLE 1-2: PIC16F874 AND PIC16F877 PINOUT DESCRIPTION

Pin Name	DIP Pin#	PLCC Pin#	QFP Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	13	14	30	I	ST/CMOS[4]	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	14	15	31	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
$\overline{\text{MCLR}}$/VPP	1	2	18	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device.
						PORTA is a bi-directional I/O port.
RA0/AN0	2	3	19	I/O	TTL	RA0 can also be analog input0.
RA1/AN1	3	4	20	I/O	TTL	RA1 can also be analog input1.
RA2/AN2/VREF-	4	5	21	I/O	TTL	RA2 can also be analog input2 or negative analog reference voltage.
RA3/AN3/VREF+	5	6	22	I/O	TTL	RA3 can also be analog input3 or positive analog reference voltage.
RA4/T0CKI	6	7	23	I/O	ST	RA4 can also be the clock input to the Timer0 timer/counter. Output is open drain type.
RA5/$\overline{\text{SS}}$/AN4	7	8	24	I/O	TTL	RA5 can also be analog input4 or the slave select for the synchronous serial port.
						PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	33	36	8	I/O	TTL/ST[1]	RB0 can also be the external interrupt pin.
RB1	34	37	9	I/O	TTL	
RB2	35	38	10	I/O	TTL	
RB3/PGM	36	39	11	I/O	TTL	RB3 can also be the low voltage programming input.
RB4	37	41	14	I/O	TTL	Interrupt-on-change pin.
RB5	38	42	15	I/O	TTL	Interrupt-on-change pin.
RB6/PGC	39	43	16	I/O	TTL/ST[2]	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock.
RB7/PGD	40	44	17	I/O	TTL/ST[2]	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as an external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured as general purpose I/O and a TTL input when used in the Parallel Slave Port mode (for interfacing to a microprocessor bus).
4: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

 © 2001 Microchip Technology Inc.

APPENDIX C

LM35 Data Sheet

National Semiconductor

December 1994

LM35/LM35A/LM35C/LM35CA/LM35D Precision Centigrade Temperature Sensors

General Description

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 μA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface mount small outline package and a plastic TO-202 package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 μA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Connection Diagrams

TO-46
Metal Can Package*

TL/H/5516-1

*Case is connected to negative pin (GND)

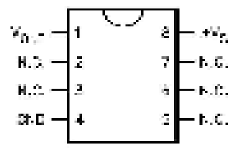
Order Number LM35H, LM35AH, LM35CH, LM35CAH or LM35DH
See NS Package Number H03H

TO-92
Plastic Package

TL/H/5516-2

Order Number LM35CZ, LM35CAZ or LM35DZ
See NS Package Number Z03A

SO-8
Small Outline Molded Package

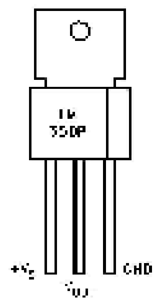


TL/H/5516-21

Top View
N.C. = No Connection

Order Number LM35DM
See NS Package Number M08A

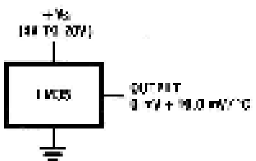
TO-202
Plastic Package



TL/H/5516-24

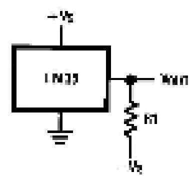
Order Number LM35DP
See NS Package Number P03A

Typical Applications



TL/H/5516-3

FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)



TL/H/5516-4

Choose $R_1 = -V_S/50\ \mu A$

$V_{OUT} = +1{,}500\ mV$ at $+150°C$
$= +250\ mV$ at $+25°C$
$= -550\ mV$ at $-55°C$

FIGURE 2. Full-Range Centigrade Temperature Sensor

TRI-STATE® is a registered trademark of National Semiconductor Corporation.

©1995 National Semiconductor Corporation TL/H/5516

Absolute Maximum Ratings (Note 10)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage		+35V to −0.2V
Output Voltage		+6V to −1.0V
Output Current		10 mA
Storage Temp.;	TO-46 Package,	−60°C to +180°C
	TO-92 Package,	−60°C to +150°C
	SO-8 Package,	−65°C to +150°C
	TO-202 Package,	−65°C to +150°C

Lead Temp.:

TO-46 Package, (Soldering, 10 seconds)	300°C
TO-92 Package, (Soldering, 10 seconds)	260°C
TO-202 Package, (Soldering, 10 seconds)	+230°C

SO Package (Note 12):

Vapor Phase (60 seconds)	215°C
Infrared (15 seconds)	220°C
ESD Susceptibility (Note 11)	2500V

Specified Operating Temperature Range: T_{MIN} to T_{MAX} (Note 2)

LM35, LM35A	−55°C to +150°C
LM35C, LM35CA	−40°C to +110°C
LM35D	0°C to +100°C

Electrical Characteristics (Note 1) (Note 6)

Parameter	Conditions	LM35A Typical	LM35A Tested Limit (Note 4)	LM35A Design Limit (Note 5)	LM35CA Typical	LM35CA Tested Limit (Note 4)	LM35CA Design Limit (Note 5)	Units (Max.)
Accuracy (Note 7)	$T_A=+25°C$	±0.2	±0.5		±0.2	±0.5		°C
	$T_A=-10°C$	±0.3			±0.3		±1.0	°C
	$T_A=T_{MAX}$	±0.4	±1.0		±0.4	±1.0		°C
	$T_A=T_{MIN}$	±0.4	±1.0		±0.4		±1.5	°C
Nonlinearity (Note 8)	$T_{MIN}\leq T_A\leq T_{MAX}$	**±0.18**		**±0.35**	**±0.15**		**±0.3**	°C
Sensor Gain (Average Slope)	$T_{MIN}\leq T_A\leq T_{MAX}$	**+10.0**	**+9.9, +10.1**		**+10.0**		**+9.9, +10.1**	mV/°C
Load Regulation (Note 3) $0\leq I_L\leq 1$ mA	$T_A=+25°C$	±0.4	±1.0		±0.4	±1.0		mV/mA
	$T_{MIN}\leq T_A\leq T_{MAX}$	**±0.5**		**±3.0**	**±0.5**		**±3.0**	mV/mA
Line Regulation (Note 3)	$T_A=+25°C$	±0.01	±0.05		±0.01	±0.05		mV/V
	$4V\leq V_S\leq 30V$	**±0.02**		**±0.1**	**±0.02**		**±0.1**	mV/V
Quiescent Current (Note 9)	$V_S=+5V$, +25°C	56	67		56	67		μA
	$V_S=+5V$	**105**		**131**	**91**		**114**	μA
	$V_S=+30V$, +25°C	56.2	68		56.2	68		μA
	$V_S=+30V$	**105.5**		**133**	**91.5**		**116**	μA
Change of Quiescent Current (Note 3)	$4V\leq V_S\leq 30V$, +25°C	0.2	1.0		0.2	1.0		μA
	$4V\leq V_S\leq 30V$	**0.5**		**2.0**	**0.5**		**2.0**	μA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.5**	**+0.39**		**+0.5**	μA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, $I_L=0$	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	$T_J=T_{MAX}$, for 1000 hours	±0.08			±0.08			°C

Note 1: Unless otherwise noted, these specifications apply: $-55°C\leq T_J\leq +150°C$ for the LM35 and LM35A; $-40°\leq T_J\leq +110°C$ for the LM35C and LM35CA; and $0°\leq T_J\leq +100°C$ for the LM35D. $V_S=+5Vdc$ and $I_{LOAD}=50$ μA, in the circuit of *Figure 2*. These specifications also apply from +2°C to T_{MAX} in the circuit of *Figure 1*. Specifications in **boldface** apply over the full rated temperature range.

Note 2: Thermal resistance of the TO-46 package is 400°C/W, junction to ambient, and 24°C/W junction to case. Thermal resistance of the TO-92 package is 180°C/W junction to ambient. Thermal resistance of the small outline molded package is 220°C/W junction to ambient. Thermal resistance of the TO-202 package is 85°C/W junction to ambient. For additional thermal resistance information see table in the Applications section.

Electrical Characteristics (Note 1) **(Note 6)** (Continued)

Parameter	Conditions	LM35 Typical	LM35 Tested Limit (Note 4)	LM35 Design Limit (Note 5)	LM35C, LM35D Typical	LM35C, LM35D Tested Limit (Note 4)	LM35C, LM35D Design Limit (Note 5)	Units (Max.)
Accuracy, LM35, LM35C (Note 7)	$T_A=+25°C$	±0.4	±1.0		±0.4	±1.0		°C
	$T_A=-10°C$	±0.5			±0.5		±1.5	°C
	$T_A=T_{MAX}$	±0.8	±1.5		±0.8		±1.5	°C
	$T_A=T_{MIN}$	±0.8		±1.5	±0.8		±2.0	°C
Accuracy, LM35D (Note 7)	$T_A=+25°C$				±0.6	±1.5		°C
	$T_A=T_{MAX}$				±0.9		±2.0	°C
	$T_A=T_{MIN}$				±0.9		±2.0	°C
Nonlinearity (Note 8)	$T_{MIN}\le T_A\le T_{MAX}$	**±0.3**		**±0.5**	**±0.2**		**±0.5**	°C
Sensor Gain (Average Slope)	$T_{MIN}\le T_A\le T_{MAX}$	**+10.0**	**+9.8, +10.2**		**+10.0**		**+9.8, +10.2**	mV/°C
Load Regulation (Note 3) $0\le I_L\le 1$ mA	$T_A=+25°C$	±0.4	±2.0		±0.4	±2.0		mV/mA
	$T_{MIN}\le T_A\le T_{MAX}$	**±0.5**		**±5.0**	**±0.5**		**±5.0**	mV/mA
Line Regulation (Note 3)	$T_A=+25°C$	±0.01	±0.1		±0.01	±0.1		mV/V
	$4V\le V_S\le 30V$	**±0.02**		**±0.2**	**±0.02**		**±0.2**	mV/V
Quiescent Current (Note 9)	$V_S=+5V$, $+25°C$	56	80		56	80		μA
	$V_S=+5V$	**105**		**158**	**91**		**138**	μA
	$V_S=+30V$, $+25°C$	56.2	82		56.2	82		μA
	$V_S=+30V$	**105.5**		**161**	**91.5**		**141**	μA
Change of Quiescent Current (Note 3)	$4V\le V_S\le 30V$, $+25°C$	0.2	2.0		0.2	2.0		μA
	$4V\le V_S\le 30V$	**0.5**		**3.0**	**0.5**		**3.0**	μA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.7**	**+0.39**		**+0.7**	μA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, $I_L=0$	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	$T_J=T_{MAX}$, for 1000 hours	±0.08			±0.08			°C

Note 3: Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output due to heating effects can be computed by multiplying the internal dissipation by the thermal resistance.

Note 4: Tested Limits are guaranteed and 100% tested in production.

Note 5: Design Limits are guaranteed (but not 100% production tested) over the indicated temperature and supply voltage ranges. These limits are not used to calculate outgoing quality levels.

Note 6: Specifications in **boldface** apply over the full rated temperature range.

Note 7: Accuracy is defined as the error between the output voltage and 10mv/°C times the device's case temperature, at specified conditions of voltage, current, and temperature (expressed in °C).

Note 8: Nonlinearity is defined as the deviation of the output-voltage-versus-temperature curve from the best-fit straight line, over the device's rated temperature range.

Note 9: Quiescent current is defined in the circuit of *Figure 1*.

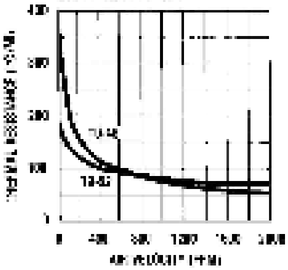
Note 10: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its rated operating conditions. See Note 1.

Note 11: Human body model, 100 pF discharged through a 1.5 kΩ resistor.

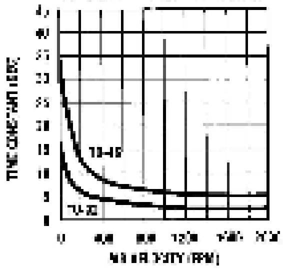
Note 12: See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" or the section titled "Surface Mount" found in a current National Semiconductor Linear Data Book for other methods of soldering surface mount devices.

Typical Performance Characteristics

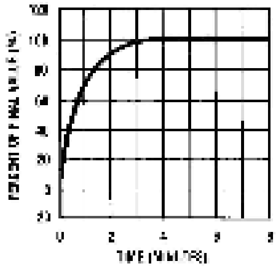
Thermal Resistance Junction to Air



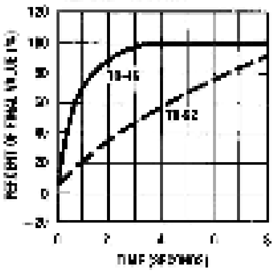
Thermal Time Constant



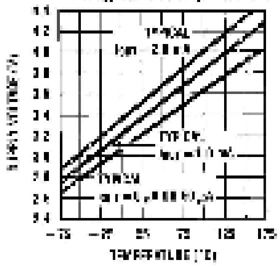
Thermal Response in Still Air



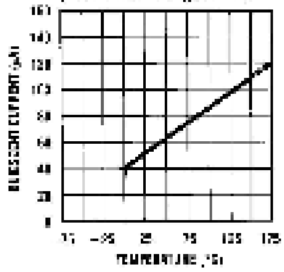
Thermal Response in Stirred Oil Bath



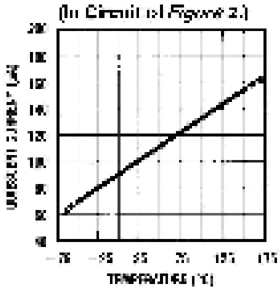
Minimum Supply Voltage vs. Temperature



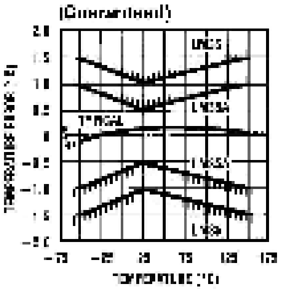
Quiescent Current vs. Temperature (In Circuit of *Figure 1*.)



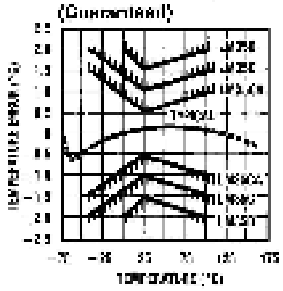
Quiescent Current vs. Temperature (In Circuit of *Figure 2*.)



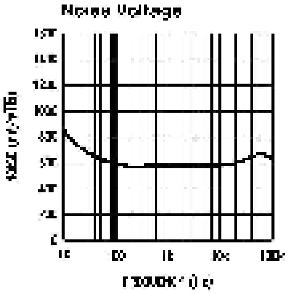
Accuracy vs. Temperature (Guaranteed)



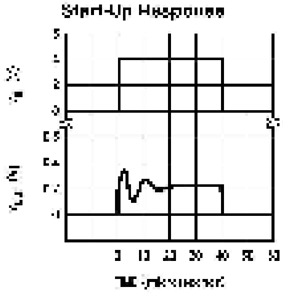
Accuracy vs. Temperature (Guaranteed)



Noise Voltage



Start-Up Response



Applications

The LM35 can be applied easily in the same way as other integrated-circuit temperature sensors. It can be glued or cemented to a surface and its temperature will be within about 0.01°C of the surface temperature.

This presumes that the ambient air temperature is almost the same as the surface temperature; if the air temperature were much higher or lower than the surface temperature, the actual temperature of the LM35 die would be at an intermediate temperature between the surface temperature and the air temperature. This is expecially true for the TO-92 plastic package, where the copper leads are the principal thermal path to carry heat into the device, so its temperature might be closer to the air temperature than to the surface temperature.

To minimize this problem, be sure that the wiring to the LM35, as it leaves the device, is held at the same temperature as the surface of interest. The easiest way to do this is to cover up these wires with a bead of epoxy which will insure that the leads and wires are all at the same temperature as the surface, and that the LM35 die's temperature will not be affected by the air temperature.

The TO-46 metal package can also be soldered to a metal surface or pipe without damage. Of course, in that case the V− terminal of the circuit will be grounded to that metal. Alternatively, the LM35 can be mounted inside a sealed-end metal tube, and can then be dipped into a bath or screwed into a threaded hole in a tank. As with any IC, the LM35 and accompanying wiring and circuits must be kept insulated and dry, to avoid leakage and corrosion. This is especially true if the circuit may operate at cold temperatures where condensation can occur. Printed-circuit coatings and varnishes such as Humiseal and epoxy paints or dips are often used to insure that moisture cannot corrode the LM35 or its connections.

These devices are sometimes soldered to a small light-weight heat fin, to decrease the thermal time constant and speed up the response in slowly-moving air. On the other hand, a small thermal mass may be added to the sensor, to give the steadiest reading despite small deviations in the air temperature.

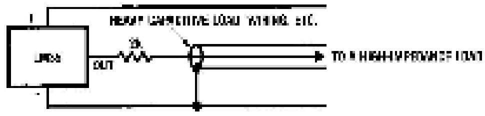
Temperature Rise of LM35 Due To Self-heating (Thermal Resistance)

	TO-46, no heat sink	TO-46, small heat fin*	TO-92, no heat sink	TO-92, small heat fin**	SO-8 no heat sink	SO-8 small heat fin**	TO-202 no heat sink	TO-202*** small heat fin
Still air	400°C/W	100°C/W	180°C/W	140°C/W	220°C/W	110°C/W	85°C/W	60°C/W
Moving air	100°C/W	40°C/W	90°C/W	70°C/W	105°C/W	90°C/W	25°C/W	40°C/W
Still oil	100°C/W	40°C/W	90°C/W	70°C/W				
Stirred oil	50°C/W	30°C/W	45°C/W	40°C/W				
(Clamped to metal,								
Infinite heat sink)	(24°C/W)				(55°C/W)		(23°C/W)	

* Wakefield type 201, or 1″ disc of 0.020″ sheet brass, soldered to case, or similar.

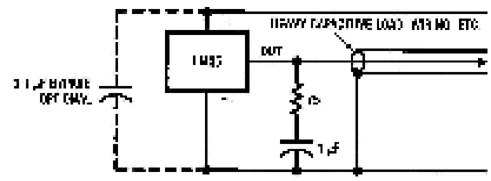
** TO-92 and SO-8 packages glued and leads soldered to 1″ square of 1/16″ printed circuit board with 2 oz. foil or similar.

Typical Applications (Continued)



TL/H/5516-19

FIGURE 3. LM35 with Decoupling from Capacitive Load



TL/H/5516-20

FIGURE 4. LM35 with R-C Damper

CAPACITIVE LOADS

Like most micropower circuits, the LM35 has a limited ability to drive heavy capacitive loads. The LM35 by itself is able to drive 50 pf without special precautions. If heavier loads are anticipated, it is easy to isolate or decouple the load with a resistor; see *Figure 3*. Or you can improve the tolerance of capacitance with a series R-C damper from output to ground; see *Figure 4*.

When the LM35 is applied with a 200Ω load resistor as shown in *Figure 5, 6, or 8*, it is relatively immune to wiring capacitance because the capacitance forms a bypass from ground to input, not on the output. However, as with any linear circuit connected to wires in a hostile environment, its performance can be affected adversely by intense electromagnetic sources such as relays, radio transmitters, motors with arcing brushes, SCR transients, etc, as its wiring can act as a receiving antenna and its internal junctions can act as rectifiers. For best results in such cases, a bypass capacitor from V_{IN} to ground and a series R-C damper such as 75Ω in series with 0.2 or 1 μF from output to ground are often useful. These are shown in *Figures 13, 14, and 16*.

Typical Applications (Continued)

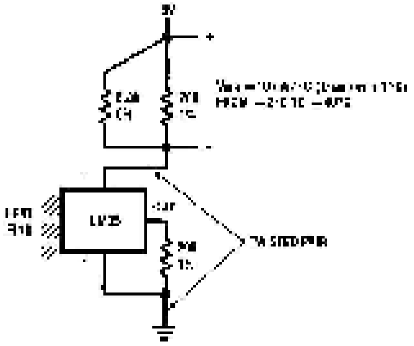


FIGURE 5. Two-Wire Remote Temperature Sensor (Grounded Sensor)

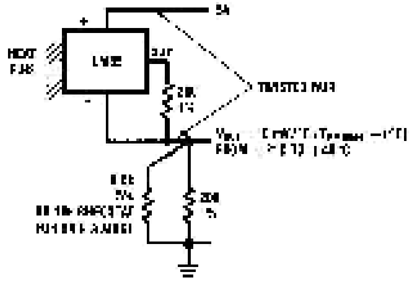


FIGURE 6. Two-Wire Remote Temperature Sensor (Output Referred to Ground)

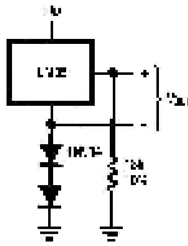


FIGURE 7. Temperature Sensor, Single Supply, −55° to 150°C

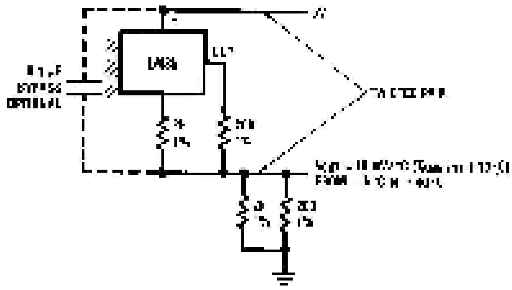


FIGURE 8. Two-Wire Remote Temperature Sensor (Output Referred to Ground)

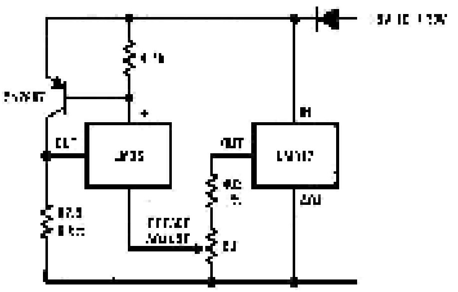


FIGURE 9. 4-To-20 mA Current Source (0°C to +100°C)

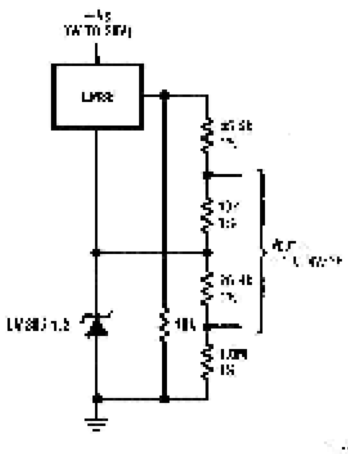


FIGURE 10. Fahrenheit Thermometer

Typical Applications (Continued)

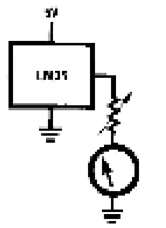


FIGURE 11. Centigrade Thermometer (Analog Meter)

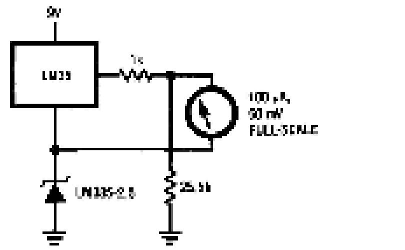


FIGURE 12. Expanded Scale Thermometer (50° to 80° Fahrenheit, for Example Shown)

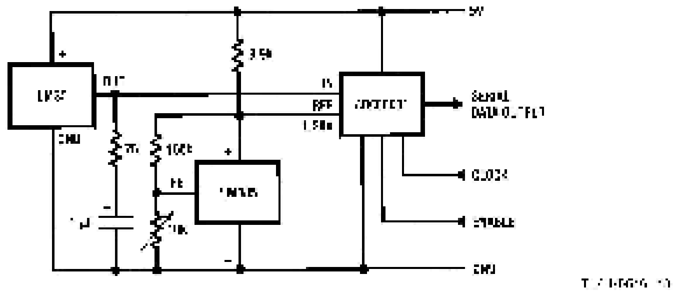


FIGURE 13. Temperature To Digital Converter (Serial Output) (+128°C Full Scale)

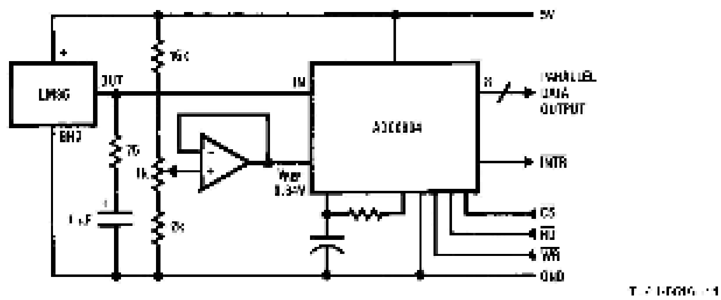
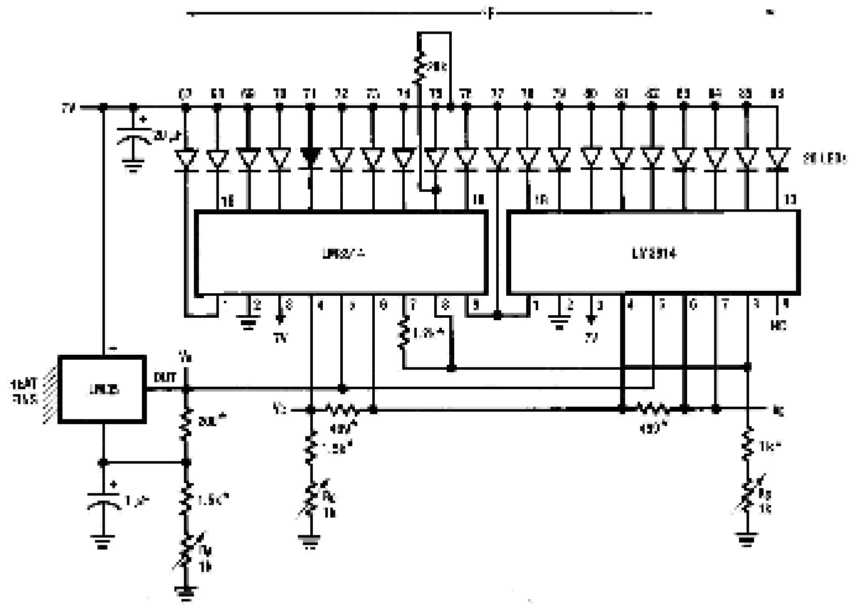


FIGURE 14. Temperature To Digital Converter (Parallel TRI-STATE® Outputs for Standard Data Bus to μP Interface) (128°C Full Scale)

Typical Applications (Continued)



*=1% or 2% film resistor
Trim RB for VB=3.075V
Trim RC for VC=1.955V
Trim RA for VA=0.075V + 100mV/°C × Tambient
Example, VA=2.275V at 22°C

FIGURE 15. Bar-Graph Temperature Display (Dot Mode)

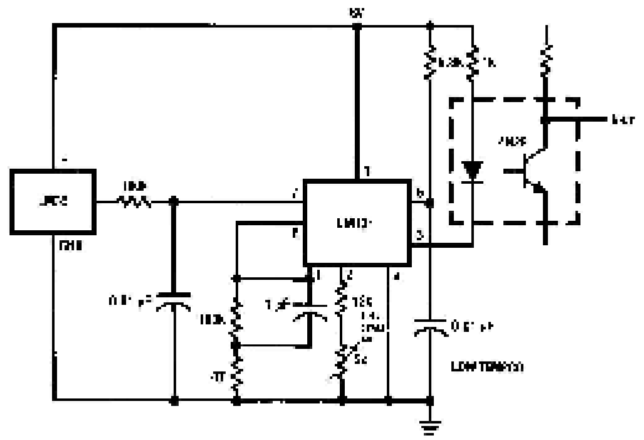
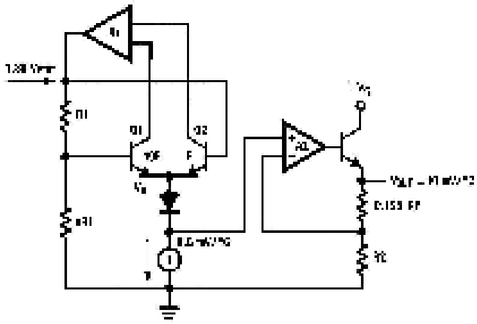
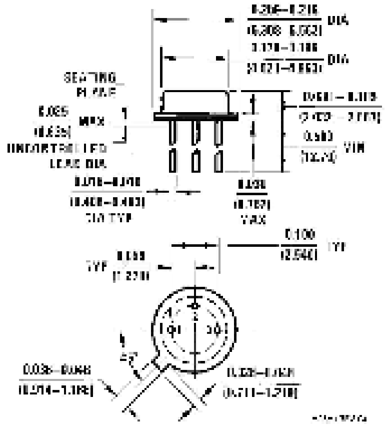


FIGURE 16. LM35 With Voltage-To-Frequency Converter And Isolated Output (2°C to +150°C; 20 Hz to 1500 Hz)

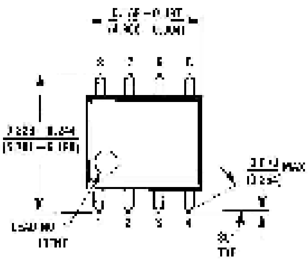
Block Diagram

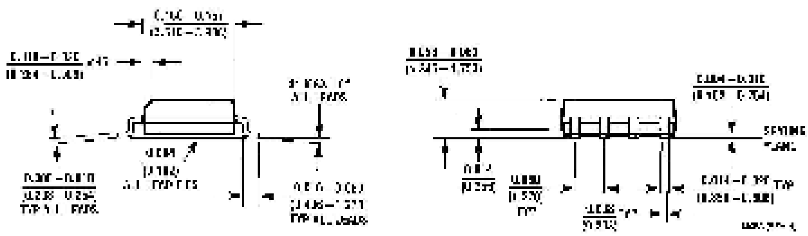


Physical Dimensions inches (millimeters)



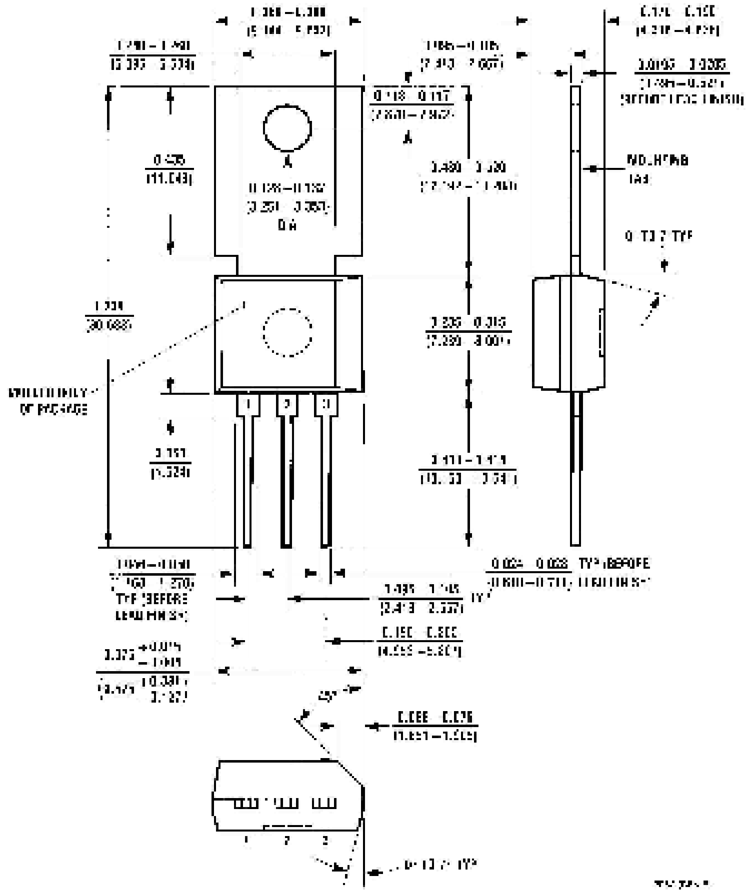
TO-46 Metal Can Package (H)
Order Number LM35H, LM35AH, LM35CH, LM35CAH, or LM35DH
NS Package Number H03H





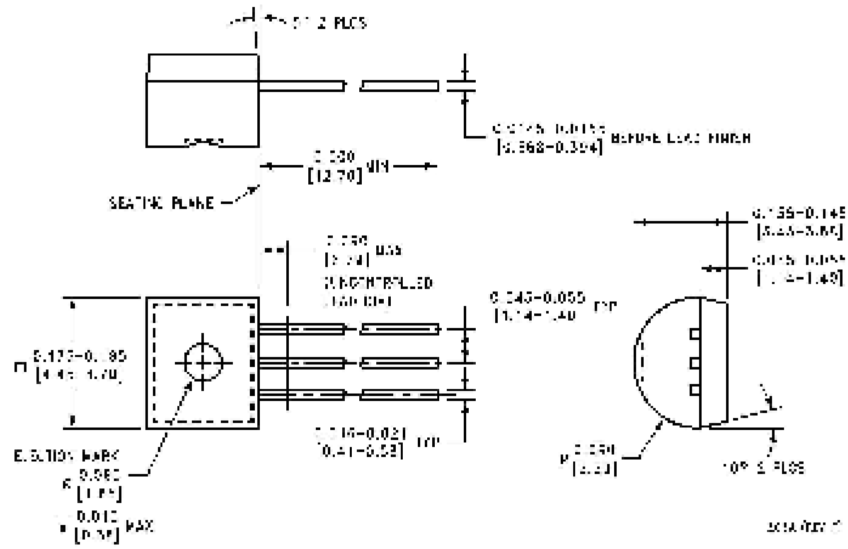
SO-8 Molded Small Outline Package (M)
Order Number LM35DM
NS Package Number M08A

Physical Dimensions inches (millimeters) (Continued)



Power Package TO-202 (P)
Order Number LM350P
NS Package Number P03A

Physical Dimensions inches (millimeters) (Continued)



TO-92 Plastic Package (Z)
Order Number LM35CZ, LM35CAZ or LM35DZ
NS Package Number Z03A

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
National Semiconductor GmbH
National Semiconductor Japan Ltd.
National Semiconductor Hong Kong Ltd.
National Semiconductores Do Brazil Ltda.
National Semiconductor (Australia) Pty, Ltd.

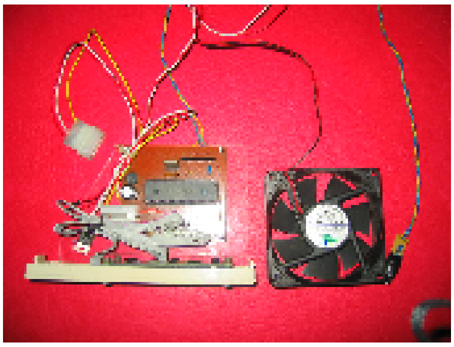
National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and National reserves the right at any time without notice to change said circuitry and specifications.

APPENDIX D

Installation Manual

Installation Manual

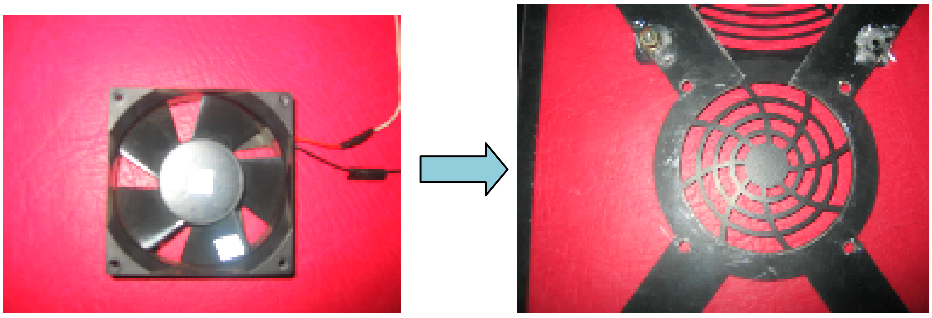
Microcontroller-Based PC Thermometer with Regulated Fan



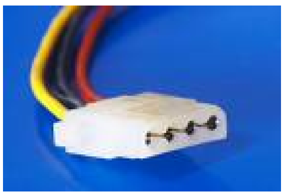
1. Select an available drive bay of the computer where the device can be mounted.

2. Place the device to the available drive bay of the computer.

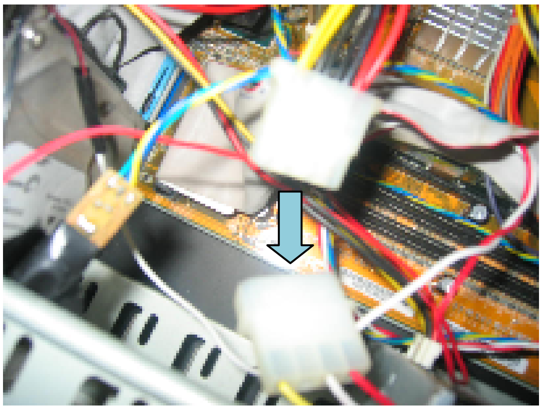
3. Make sure the locks on the side of the device would fit on the drive bay to ensure that the device is placed correctly.
4. Make sure the casing of the computer can accommodate an extra cooling fan (usually side panel of the casing). If not, the casing of the computer must be modified.
5. Install the cooling fan which is connected to the device to the casing of the computer.



6. Place the temperature sensor near the processor or heat sink where it emits high temperature inside the computer.
7. Make sure all wires and connections are not entangled.
8. Choose an available connector of the computer's power supply unit and make sure it is the right source (12V).



9. Connect the power supply cable of the device to the available connecter of the computer. Make sure the computer is turned off before connecting the device.



10. After connecting the power supply cable, observe for the display on the LCD.

11. Make sure the casing (side panel where the fan is installed) is properly screwed to ensure proper airflow inside the computer.

12. The device is now ready to use.

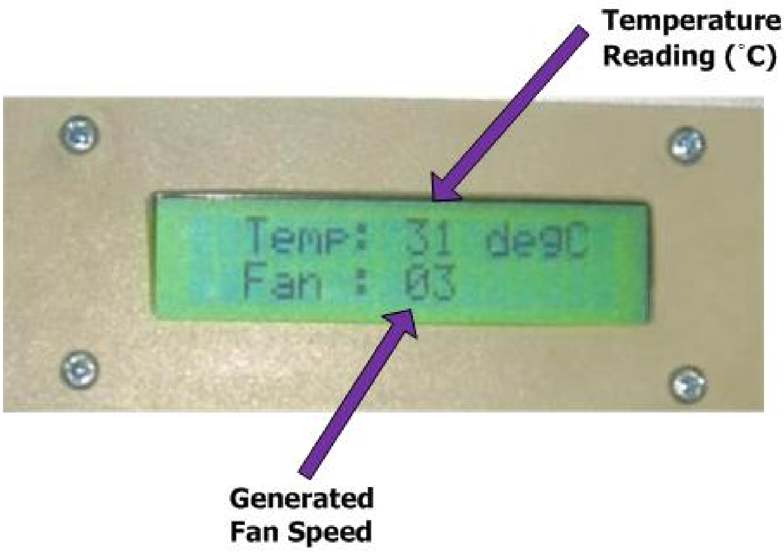
APPENDIX E

User’s Manual

User's Manual

Microcontroller-Based PC Thermometer with Regulated Fan

1. Make sure the device is properly installed by following the steps on the Installation Manual.
2. The device can be used as soon as it is installed properly.
3. The LCD panel serves as the display.



4. The temperature inside the computer can now be monitored which is displayed on the LCD panel.
5. The fan speed depending on the measured temperature is generated automatically.